WRDC-TR-89-8037

FLEXIBLE AUTOMATIC DISCRETE PARTS ASSEMBLY

D. E. Atkins
R. A. Volz

The University of Michigan
College of Engineering
Ann Arbor MI 48109-2110

September 1989

Final Report for Period September 1987 - August 1988

Manufacturing Technology Directorate
Wright Research and Development Center
Air Force Systems Command
Wright-Patterson Air Force Base, Ohio 45433-6533

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

Michael F. Hitchcock

MICHAEL F. HITCHCOCK, Chief
Implementation Branch
Integation Technology Division

FOR THE COMMANDER

Bruce A. Rasmussen

BRUCE A. RASMUSSEN, Chief
Integration Technology Division
Manufacturing Technology Directorate

If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify WRDC/MTI, WPAFB, OH 45433-6533 tp help us maintain a current mailing list.

Copies of this report should not be returned unless return is required by security considerations, contractual obligation, or notice on a specific document.

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION: Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution is unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): RSD-TR-14-88	5. MONITORING ORGANIZATION REPORT NUMBER(S): WRDC-TR-89-8037
6a. NAME OF PERFORMING ORGANIZATION: College of Engineering, The University of Michigan; 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION: Wright Research and Development Center, Manufacturing Technology Directorate (WRDC/MTI)
6c. ADDRESS (City, State, and ZIP Code): Ann Arbor, MI 48109-2110	7b. ADDRESS (City, State, and ZIP Code): Wright-Patterson AFB, OH 45433-6533
8a. NAME OF FUNDING/SPONSORING ORGANIZATION; 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: F33615-85-C-5105
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.
61102F	2306	P9	01

11. TITLE (Include Security Classification): Flexible Automatic Discrete Parts Assembly

12. PERSONAL AUTHOR(S): D.E. Atkins, R.A. Volz

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM 9/1/87 TO 8/31/88	September 1989	97

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Manufacturing Science, Robotics, Artificial Intelligence, Sensors, Machine Vision, Special Purpose Processors, Motion Planning, Knowledge Based Systems, Requirements Analysis

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This project addresses the problem of flexible automated assembly. The long term goal is to develop technologies that can be applied to many different aspects of assembly automation and that can be evolved to successively higher levels of abstraction, i.e. that will eventually allow high level commands, such as "assemble product X", to be given. We are addressing three aspects of the problem, vision, planning/programming, and system integration. This report describes the second year's effort toward these goals. Specific achievements during this reporting period are:

(CONTINUES)

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☐ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☒ DTIC USERS	UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code)	22c. OFFICE SYMBOL
Michael F. Hitchcock	513-255-7371	WRDC/MTI

19. Abstract (cont)

Highlights of Research Accomplishments

The following paragraphs briefly overview the principal accomplishments in each of the major research areas. It is important to recognize that in each of these areas there is significant additional work being done under the auspices of other contracts acquired, in part, due to the recognition and base of support provided by the Air Force.

Computer Vision

A flexible assembly system must be able to perceive its environment in order to take appropriate actions. Computer vision is being increasingly used to help machines achieve this perception. We are addressing several aspects of vision particularly important to manufacturing, including the following:

- Object recognition from partial views,
- Object recognition using range images,
- Dynamic scene analysis,
- Motion stereo,
- Qualitative vision, and
- Knowledge based vision techniques.

Specific achievements during this reporting period are:

- Our multi-featured object recognition system has been extended to incorporate a scale independent feature.
- A new technique called Feature Modulated Attraction (FMA), has been designed to extend the 2-D recognition work to 3-D and obtain position and orientation match between model and image data.
- Prior work on Ego-motion Complex Log Mapping (ECLM) was shown to maintain projection invariance for arbitrary translational motion. A new interpolation technique using a Gaussian weighted image in a 3 X 3 window that varies with distance from the origin was developed and shown to yield improved results.

(CONTINUES)

19. Abstract (Cont)

- A new algorithm for correcting uncertainty of camera motion in our depth from motion stereo system has been developed. As a result, depth values can be maintained in a camera independent coordinate frame.

- A vision workbench has been implemented in Common Lisp.

Planning and Programming

As is evident from Figure 1 there are several levels of planning and support tools that are necessary to achieve automated assembly. The areas investigated during the past year include:

- Gross motion path planning,

- Assembly sequence planning,

- Planning how to use sensor information to overcome uncertainties in geometrics (tolerances), sensor errors, and control errors,

- Heuristic problem solving, and

- Development of an Engineering *Database/Knowledge* Management System.

Accomplishments during the past year include:

- A simple probablistic measure of "workspace" that accounts for various sizes and shapes of obstacles was defined previously. A technique for partitioning the workspace has been developed that allows the search algorithm to be subdivided in a way that increases computational efficiency.

- A new learning algorithm for automatically deriving search control knowledge for planning from problem solving histories has been developed. Tests have shown that it significantly improve heuristic planning previous methods.

- An experimental system, XAP/1 has been implemented to solve assembly planning problems. It incorporates directionality, manipulability, and fixture complexity cost measures. It has been successfully tested on significant problems.

(CONTINUES)

19. Abstract (Cont)

- We have developed simple replanning strategies for dealing with failures in peg-in-the-hole assembly tasks, and have derived design constraints under which the strategies can be shown to work.

- We have developed a simple query language named ESQL tailored specifically to engineering databases. The syntax of the language has been designed and tested, and an algorithm translating ESQL algebra into a relational algebra has been designed.

Systems Integration and Architecture Techniques

Systems integration is essential for any robotics for manufacturing system, but is today done with little more than ad hoc tools, and often starting from scratch each time a system is built. Moreover, there is little experience with the use of newer forms of parallel architectures for robotics and manufacturing applications. In this area we are investigating:

- A robotics prototyping system,

- The mapping of (vision) algorithms unto parallel architectures, and

- Distributed Language techniques, which we believe to be fundamental to developing a systematic way of accomplishing systems integration.

Specific accomplishments during the past year include:

- The workstation-based rapid prototyping "vision workbench" developed in previous years has been enhanced to support our two PUMA robots and the Mercury ZIP array processor.

- Several algorithms for parallelizing branch and bound techniques for low level vision applications were tested on a 64 node hypercube processor. A load balancing algorithm was developed.

- Our Distributed Ada Translation system has been extended to allow remote elaboration of task object from task types and the instantiation of
generics on any node in the system.

EXECUTIVE SUMMARY

Motivation and Goals

There is great emphasis today on the development of fully integrated manufacturing, including both the automation of individual manufacturing operations and the integration of all phases of manufacturing, from design through planning through production. The associated problems, however, are extremely complex and will require a large research effort over and extended period of time. This project is focused on an important category of flexible automation problems, in itself quite large, for which there are yet very few operational systems, flexible automated assembly. Assembly is the most highly labor intensive manufacturing process in the production of durable goods. There is thus great potential for direct cost saving through the development of flexible automated systems for assembly, as well as the indirect benefits mentioned above.

The long term goal of this work is to develop technologies that can be applied to many different aspects of assembly automation and that can be evolved to successively higher levels of abstraction, i.e., that will eventually allow high level commands, such as "assemble product X", to be given. We are addressing three aspects of the problem, vision, planning/programming, and system integration. This report describes the first year's effort on this project toward these goals.

Context of Research

Figure E.1 portrays a taxonomy of an automated assembly system into which our work fits. The boxes represent capabilities and operations that must be developed while inputs and outputs are shown in ovals. The vertical sequence of operations and outputs proceeding downward in the middle of the diagram illustrates the sequence of operations that must be performed in order to automatically assemble a product. The other boxes indicate essential capabilities which must be available in order to achieve the operation shown in the center. Few of the components shown are adequately developed at present. Moreover, many of the different constituents in an automatic assembly system will be implemented on different computers and, possibly, in different languages. Integration of these constituents into a working whole is an extremely important aspect of the overall problem.

The distinction between off-line and on-line activities is somewhat variable among different systems, but as more sophisticated and complex planning algorithms are developed will tend to move upward from the position shown in the figure. Our work addresses aspects of most of the off-line operations shown, the sensor data processing portion, primarily vision, of the on-line processing and the systems integration techniques by which such a system can be made operational. Our approach is to use existing technology, where available, coordinate

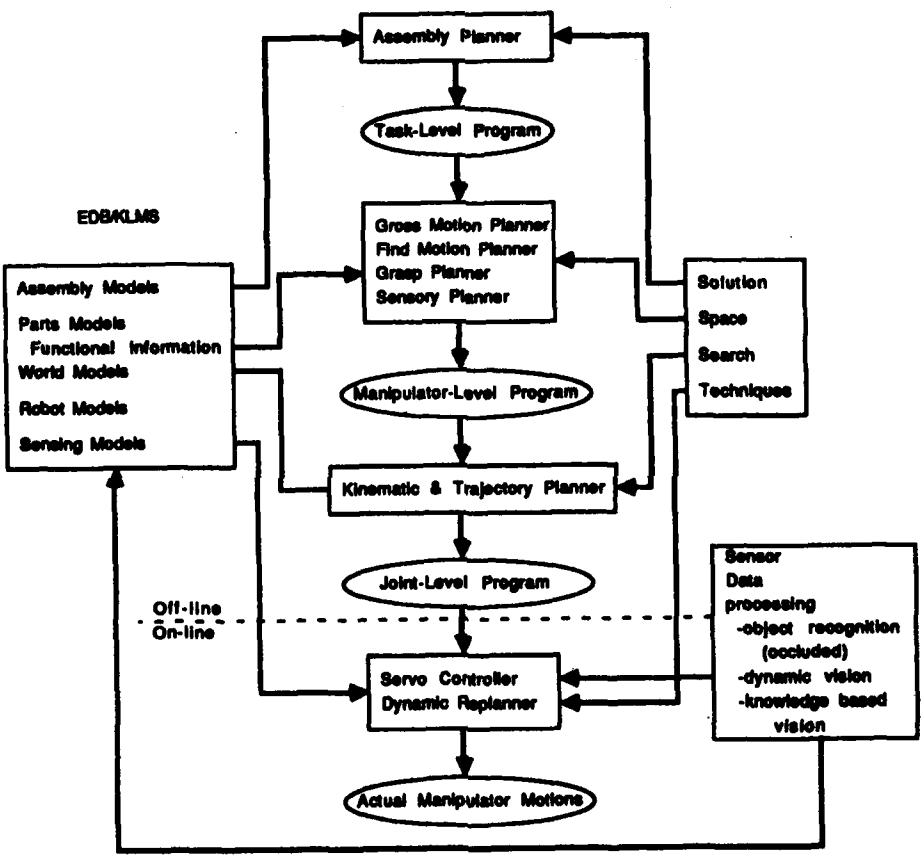


Figure E.1: Interrelation between components in an automated assembly system

with other research projects in the Robotics Research Laboratory and focus the research on this project toward key underlying technologies.

Highlights of Research Accomplishments

The following paragraphs briefly overview the principal accomplishments in each of the major research areas. It is important to recognize that in each of these areas there is significant additional work being done under the auspices of other contracts acquired, in part, due to the recognition and base of support provided by the Air Force.

Computer Vision

A flexible assembly system must be able to perceive its environment in order to take appropriate actions. Computer vision is being increasingly used to help machines achieve this perception. We are addressing several aspects of vision particularly important to manufacturing, including the following:

- Object recognition from partial views,
- Object recognition using range images,
- Dynamic scene analysis,
- Motion stereo,
- Qualitative vision, and
- Knowledge based vision techniques.

Specific achievements during this reporting period are:

- Our multi-featured object recognition system has been extended to incorporate a scale independent feature.
- A new technique called Feature Modulated Attracton (FMA), has been designed to extend the 2-D recognition work to 3-D and obtain position and orientation match between model and image data.
- Prior work on Ego-motion Complex Log Mapping (ECLM) was shown to maintain projection invariance for arbitrary translational motion. A new interpolation technique using a Gaussian weighted image in a 3×3 window that varies with distance from the origin was developed and shown to yield improved results.

- A new algorithm for correcting uncertainty of camera motion in our depth from motion stereo system has been developed. As a result, depth values can be maintained in a camera independent coordinate frame.
- A vision workbench has been implemented in Common Lisp.

Planning and Programming

As is evident from Figure 1 there are several levels of planning and support tools that are necessary to achieve automated assembly. The areas investigated during the past year include:

- Gross motion path planning,
- Assembly sequence planning,
- Planning how to use sensor information to overcome uncertainties in geometries (tolerances), sensor errors, and control errors,
- Heuristic problem solving, and
- Development of an Engineering Database/Knowledge Management System.

Accomplishments during the past year include:

- A simple probabilistic measure of "workspace" that accounts for various sizes and shapes of obstacles was defined previously. A technique for partitioning the workspace has been developed that allows the search algorithm to be subdivided in a way that increases computational efficiency.
- A new learning algorithm for automatically deriving search control knowledge for planning from problem solving histories has been developed. Tests have shown that it significantly improve heuristic planning previous methods.
- An experimental system, XAP/1 has been implemented to solve assembly planning problems. It incorporates directionality, manipulability and fixture complexity cost measures. It has been successfully tested on significant problems.
- We have developed simple replanning strategies for dealing with failures in peg-in-the-hole assembly tasks, and have derived design constraints under which the strategies can be shown to work.

- We have developed a simple query language named ESQL tailored specifically to engineering databases. The syntax of the language has been designed and tested, and an algorithm translating ESQL algebra into a relational algebra has been designed.

Systems Integration and Architecture Techniques

Systems integration is essential for any robotics or manufacturing system, but is today done with little more than ad hoc tools, and often starting from scratch each time a system is built. Moreover, there is little experience with the use of newer forms of parallel architectures for robotics and manufacturing applications. In this area we are investigating:

- A robotics rapid prototyping system,
- The mapping of (vision) algorithms unto parallel architectures, and
- Distributed Language techniques, which we believe to be fundamental to developing a systematic way of accomplishing systems integration.

Specific accomplishments during the past year include:

- The workstation-based rapid prototyping "vision workbench" developed in previous years has been enhanced to support our two PUMA robots and the Mercury ZIP array processor.
- Several algorithms for parallelizing branch and bound techniques for low level vision applications were tested on a 64 node hypercube processor. A load balancing algorithm was developed.
- Our Distributed Ada Translation system has been extended to allow remote elaboration of task object from task types and the instantiation of generics on any node in the system.

Organizational Considerations

The third year has seen a continued excellent performance in terms of student participation and technical publication. Summary statistics directly supported by Air Force funding are:

- 20 papers presented at major conferences,
- 11 papers appeared in reviewed journals,

- 3 papers are currently under review or have been accepted for journal publication or conference presentation.
- 12 graduate students participated in the program,
- 3 students participating in the program received the Ph.D.

Contents

List of Figures

1 Introduction

1.1 Motivation and Goals

There is great emphasis today on the development of fully intergated manufacturing. Indeed, the development of integrated manufacturing techniques is critical to the survival of many segments of United States' industry. Integrated manufacturing includes both the automation of individual manufacturing operations and the integration of all phases of manufacturing, from design through planning through production. The motivations for integrated manufacturing, ultimately, are economic, though there are numerous intermediate reasons such as improved quality, reduced lead time on introducing new products, reduced inventories (through the ability to quickly produce new parts when needed) and the ability to rapidly prototype new designs, in addition to the traditional reason of reduced direct production cost.

This project is focused on an important category of flexible automation problems for which there are very few operational systems, flexible automated assembly. Assembly is the most highly labor intensive manufacturing process in the production of durable goods [1]. There is thus great potential for direct cost saving through the development of flexible automated systems for assembly, as well as the indirect benefits mentioned above.

There are many levels of assembly automation one can seek. Ultimately one would like to be able to give a very high level command such as "Build 30 of product X by deadline," and have the system automatically schedule the assembly process, order the components and tools needed, schedule their delivery to the work area, determine the assembly sequence and fixturing required, generate the robot programs required, and manage the flow of all material and information to accomplish the assembly. The achievement of this level of automatic operation is a long way off. A slightly lower, but also very difficult level is the task level, in which one would like to be able to give commands such as "assemble product X," "insert bolt A in hole B," or "grasp bracket C" under the assumption that fixturing, materials, and tooling are all available inputs; even within this level there are obviously many sublevels.

The long term goal of our work is to develop technologies that can be applied to any of several levels of assembly automation and can be evolved to higher levels of abstraction. In particular we are addressing three aspects of the problem, vision, planning/programming, and system integration.

1.2 Overview of Automatic Assembly

Figure 1 portrays the major components which must be part of an automated assembly system at the task level (i.e., the "assembly product X" level), and the relationships between

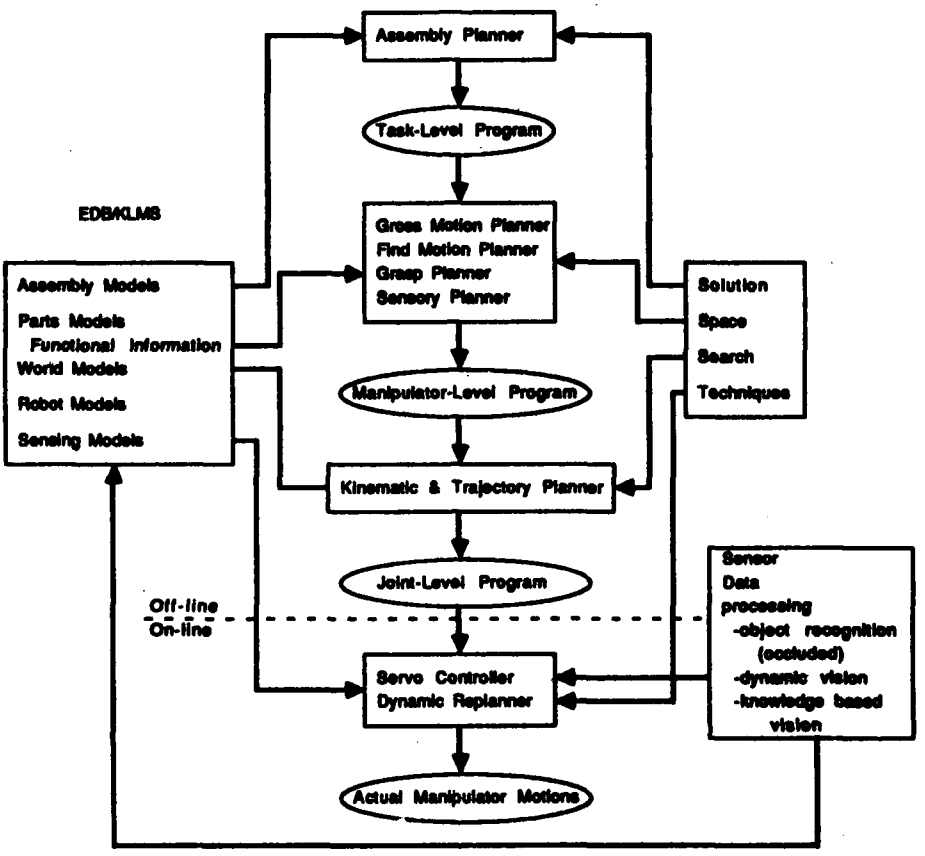


Figure 1: EDB/KLMS

them. The boxes represent capabilities and operations while inputs and outputs are shown in the ovals. The vertical sequence of operations and outputs proceeding downward in the middle of the diagram illustrates the sequence of operation which must be performed in order to automatically assembly a product. The other boxes indicate essential capabilities which must be available in order to achieve the operation shown in the center. Few of the components shown are adequately developed at present. Indeed, even the appropriate contents of the database are not fully known at this time.

We assume as inputs to the process a wide variety of design, work cell model and sensor information. In order to manage this information, an Engineering Database/Knowledge Management System (EDB/KMS) is necessary that combines the data storage/retrieval capabilities normally present in database management system with semantics extraction and

inferential mechanisms normally associated with knowledge management systems. Further, the EDB/KMS must support access at various levels of abstraction with corresponding levels of access times.

The assembly planner produces the sequence in which the components to the product are to be assembled, and the associated fixturing and orientation information for each step in the sequence, called the (a subtask level program) in Figure 1. Each step at the subask level consists of a single operation such as placing a part in a fixture, changing a tool on a robot end effector, or mating two parts.

Typically there is a robot motion associated with each subtask step resulting from the assembly sequence planner. Each such subtask requires several further stages of planning. First, the motion form the initial position of the robot to a position in free space near its final position must be determined which avoids collision and minimizes an appropriate cost function, e.g., time or energy (gross motion planning). Next motions involving contact among the parts being mated must be planned (fine motion planning). These two planning stages are separated because some gross motion planners [2] do not permit contact (though in some other cases the two might be performed jointly [3]. Also, if a component is to be picked up by a robot, the point at which it is to be grasped must be chosen so that no interference occurs among the hand, the component and other parts of the assembly, and also so that the component will not slip in the grasp of the robot. Finally, at this stage of planning, sensor plans must be developed which can use the robot's sensors to determine whether or not a subtask step was successfully completed.

From the perspective of the robots, the output of the subtask planning stage is a sequence of specific motion, gripper and sensor commands in a world coordinate system of the workspace. The kinematic and trajectory planner converts these motions into joint motions of the robots involved and determines the planned time history of each joint along the desired path.

The servo level, of course, directs the movement of each joint of the robots. It will be important for assembly tasks that the control system be capable of compliant motion (e.g., move in a given direction while maintaining a prescribed contact force). Guarded motions, i.e., motions that continue until some sensed force, torque, or position condition is satisfied, will also be important and, indeed, are closely related to the sensor planning for subtask verification. Essentially, as each guard condition satisfied and a motion stopped, the sensor readings are compared with planned values to determine whether or not the desired motion actually occurred. Unfortunately, uncertainties in part geometries, actual gripping position, sensor errors, control errors, etc., fairly frequently result in a motion being stopped due to excessive force or torque. When this is detected, as a result of the sensor verification plan, a dynamic fine motion replanning is necessary. Since presumably the robot is close to the desired position, this planner can be much simpler, and hence done in real-time, than the

original planner.

Several of the planning stages required searching extensive solution spaces. Efficient techniques are necessary if automatic planning is to be realistically achieved.

It is obvious that a multitude of sensing is required for successful operation of an assembly system. Vision, force/torques, tactile and range sensors are the primary candidates, though if the assembly involves are welding, temperature sensing (of the weld puddle) may prove useful. Many of the actual motion target positions will be determined from sensor information, some servoing of the motions may be driven by higher level sensors (e.g., vision) and certainly sensing is required for verification.

Finally, many of the different constituents in an automatic assembly system will be implemented on different computers and, possibly, in different languages. Integration of these constituents into a working whole is an extremely important aspect of the overall problem.

1.3 Project Organization and Activities

Fully automated discrete parts assembly is a very complex problem and will require a large research effort over an extended period of time. Our approach is to use existing technology, where available, coordinate with other research projects in the Robotics Research Laboratory and direct the research on this project toward key underlying technologies. This concept is illustrated in Figure 2. In particular, our research emphasis may be grouped in three important categories, machine vision and sensing, programming and planning, and systems integration techniques. Figure 3 shows the organization of the project in greater detail, and identifies the major subprojects. Due to budgetary cutbacks, activity in the area shown in dotted lines was curtailed.

The machine vision research is exploring a variety of techniques ultimately useful for object recognition and pose determination. This is being studied via single image analysis, the analysis of a sequence of images, and the use of knowledge based approaches and with both grey level and range images as inputs. Results obtained for occluded part recognition have been shown to outperform previously known techniques, the fundamental basis for differential geometric descriptions and segmentation of surfaces in range images has been established, and promising work has been started on the mapping of vision algorithms onto parallel processors of a hypercube architecture.

Work in the planning and programming area has produced significant results in path planning, heuristic problem solving techniques, and assembly and sensor planning. New, more efficient, path planning algorithms that take into account size, shape and orientation of obstacles, and develop a probability measure of reaching a deadend during path searching

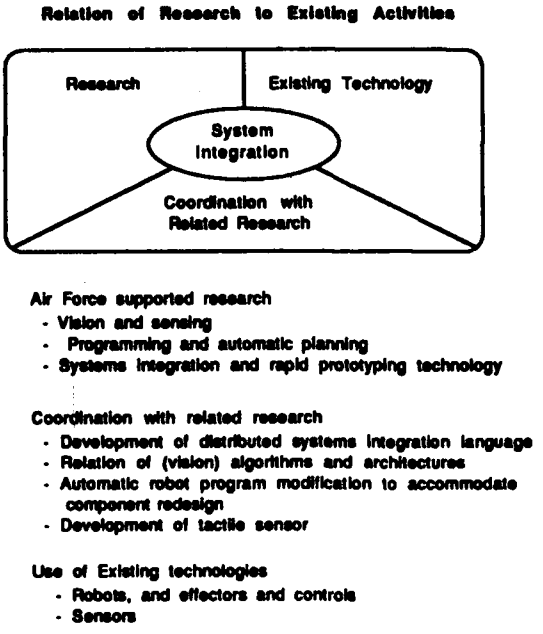


Figure 2: Relation of Research to Existing Activities

have been developed. In the area of general problem solving, the techniques obtained have been shown to be very robust, exhibiting performance equal to the best individual domain specific solution obtained across a broad range of problems.

To date there has been almost no work on automatic planning of assembly sequences. We are initially approaching the problem by considering a restricted class of assembly tasks involving primarily axis oriented operations, i.e., placing parts (e.g. brackets, washers, nuts) on a shaft of some type. New methods have been developed for representing geometric feasibility of an assembly sequence and searching the feasibly space to minimize assembly costs according to the models of assembly developed earlier. In the area of sensor planning we are introducing two important new principals: 1) establishing relations between assembly design constraints and the use of sensors to guide the assembly operation in the present of uncertainties, and 2) the reduction of the problem dimensionality through the use of *contact formations* as the basis for verification and replanning rather than six dimensional configurations. It has been shown that it is possible, under suitable design constraints, to identify the contact formation in which a pair of parts lie (except for "don't care" situations in which the same replanning strategy may be applied) even the the presence of uncertainties, and that simple on-line replanning strategies can often be used to correct errors that arise.

The systems integrations work has had two aspects: 1) building a rapid prototyping

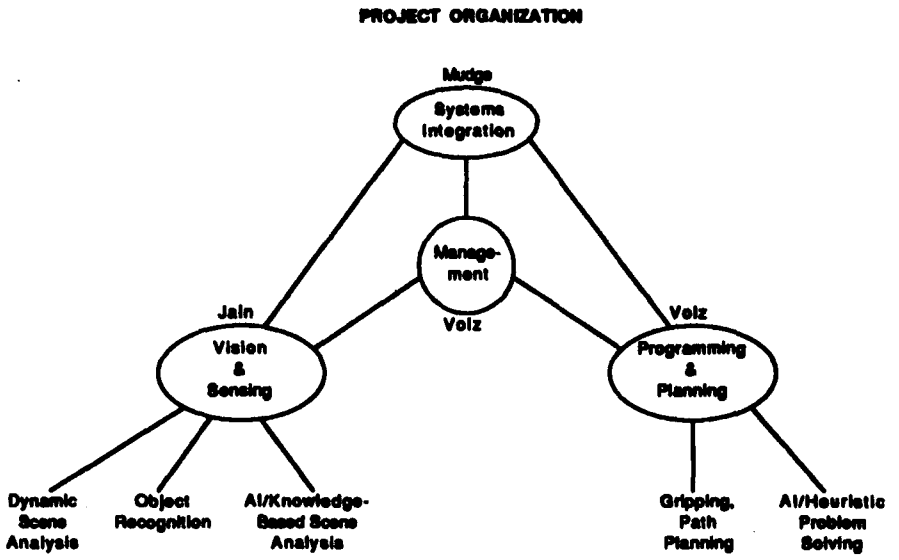


Figure 3: Project Organization

robot/vision system which can be used as the basis for future experimentation and research, and 2) serving as a focusing agent for systems integration techniques based around distributed languages (largely supported by other contracts). The rapid prototyping system is fully operational. Image acquisition and edge detection essentially proceeds in real-time, making the complete real-time implementation of vision algorithms a real possibility in our future work. Moreover, it can be accessed and used over the campus computer network. In view of the importance of and need for systems integration techniques for distributed systems and the role of this project as a focusing agent for our work in this area, our vision of distributed manufacturing software will be described and related activities described briefly.

References

[1] A.G. Boothroyd. "Use of Robots in Assembly Automation," *Annals of the CRP*, Vol. 3, No. 2, 1984.

[2] E.G. Gilbert and D.W. Johnson. "Distance Functions and their Application to Robot Path Planning," *IEEE Journal of Robotics and Automation*, Vol. RA-1, No. 1, pp. 21-30, March 1985.

[3] Tomas Lozano-Perez, and M.T. Mason and R.H. Taylor. "Automatic Synthesis of Fine Motion Strategies for Robots," *The Inter. Jour. of Robotics Research,* Vol. 3, No. 1, pp 3-24, 1984.

2 Research

2.1 Computer Vision

2.1.1 Range Image Understanding

Research Objective

In most applications, a 3-dimensional object must be recognized from its 2-dimensional projections. A complete object recognition system must have a geometric database to which objects that are being recognized can be compared. A CAD database is an excellent source for such a geometric database. CAD databases generally contain view-independent descriptions of objects, usually consisting of equations describing each surface of an object. However, CAD object model does not give direct information about the object appearances and which features belong to an object. This makes comparison of extracted features in the image with surface descriptions in object model a very difficult task. We need an intermediate object representation which is more suitable for object recognition purpose. This representation will be in the form of an aspect graph, which specifies how the features and their relationships change as the object is seen from different viewpoints.

Status of the Project

In a scene, there may be several objects that may be only partially visible. Global features cannot be used to recognize objects from their partial views. Local features have been proposed for recognizing objects. The success of these has been limited due to the difficulties in their determination and matching.

A symbolic surface descriptor will capture the intrinsic nature of the surface as a global feature. Such a descriptor will be very useful in object recognition if it can be computed from partial views of object surfaces. Commonly used global features in object recognition lack this desirable characteristic. We are trying to develop the concept of symbolic surface descriptors for object recognition in range images [3] using the surfaces given by our segmentation algorithm [2, 3].

In the last year, our efforts were focused more on generating aspect graphs. The object may look completely different from one viewpoint when compared with its appearance from a second viewpoint. When we examine the projection of a 3-dimensional object on the image plane, we find that from most viewpoints, a small change in the viewing direction results in a small change in the image. The qualitative nature of the projected images before and after the change remains identical: the number and types of symbolic surface descriptors of one view is the same as another view. However, if we progressively change our viewpoint, at some point we will experience a sudden change in the qualitative appearance of the object.

This is called an event. These changes are due to the creation or annihilation of features, or to an alteration in the configuration. Thus, if we view a 3-dimensional object from all possible directions, there will be a finite number of qualitatively distinct appearances of the object. Each such qualitative view of the object is called an aspect. Each aspect of an object can be seen from certain open subsets of viewing directions. Transition from one aspect to another corresponds to an event which will occur when we traverse from one subset of viewpoints to the other. All possible qualitative appearances of an object can be represented by a graph structure, known as an aspect graph. Each node gives information about a configuration of symbolic surface descriptors and each arc a description of the changes that will occur during transition between adjacent nodes. Using the aspect graph, it is now possible to determine which symbolic surface descriptors belong to an object. This information can be used to derive a plan for recognition. Matching of the features in the model with those in an image and verification of an image interpretation can be achieved more easily since the object model and images have been reduced to the same symbolic level.

We are now studying the problem of how to obtain aspect graphs from the CAD database. Clearly, an exhaustive search that examines the projections of an object from all possible viewpoints in order to determine possible aspects of the object is not a desirable method. The best way to generate an aspect graph is to directly calculate the sets of viewpoints from each of which we perceive an unique aspect of an object. This can be done by computing the viewpoints where the aspect changes occur. These viewpoints corresponding to events serve as the boundaries among sets of viewpoints associated with object's aspects. After obtaining these boundaries partitioning the space of possible viewpoints, an aspect graph can be easily determined.

There are a few types of events, each type of event gives rise to certain changes in feature structure. In general, aspect changes are due to changes in visibilities, nature of surfaces, and occluding relations among them. The clear distinctions between two aspects are the number of symbolic surface descriptors and spatial relationships among them on the image. Surface normal characteristics are used to determine from which range of viewpoints a given surface patch is visible or partially visible. Geometric relationships between surfaces described in a CAD object model allow us to find out the occluding relations among surfaces as seen by the observer and how they are changed as we change the viewpoint. There are strong relationships between intrinsic nature of surfaces and the viewpoints where each event occurs. Mathematical equations to compute the viewpoints where object appearance suddenly changes, due to changes in surface visibilities and occluding structure among them, have been developed. These equations relate the descriptions of surfaces in a CAD object model to the event occurrences. The mathematical tools employed in our approach are theorems from Differential Geometry and Singularity Theory. Our method is applicable to both polyhedral objects and curved objects. We are in the process of implementing the algorithm to generate an aspect graph based on this idea.

Future Research

Our future research in aspect graph generation is to study how and when an aspect of object is changed due to changes in types of symbolic surface descriptors. Though a given surface patch may be partially visible from two different viewpoints, we may perceive two different types of symbolic surface descriptors. This is because we see different portions of the same surfaces with different shape intrinsic natures. This problem is strongly related to the classification scheme of symbolic surface descriptors as mentioned above.

References

[1] Besl, P.J. and R. Jain, "Segmentation Through Symbolic Surface Descriptions", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 77-85, June 1986.

[2] Besl, P.J. and R. Jain, "Segmentation Through Symbolic Surface Descriptions", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 10, No. 2, pp. 167-192, March 1988.

[3] Jain, R., T. Sripradisvarakul and N. O'Brien, "Symbolic Surface Descriptors for 3-D Object Recognition", *Proc. SPIE Conf. on Optical and Digital Pattern Recognition*, Vol. 754, pp. 82-91, Jan. 1987.

2.1.2 Dynamic Scene Analysis

Research Objective

Stereo information can be obtained using a single translating camera. The analysis of a sequence of images obtained using such a camera configuration may be facilitated by using a mapping similar to the retino-striate mapping in mammals, which we call the Ego-Motion Complex Logarithmic Mapping (ECLM). The mathematical properties of this mapping make it ideal for object recognition, depth determination, and dynamic scene segmentation. The fact that this mapping facilitates analysis of a dynamic scene acquired using a moving camera makes it very attractive in many practical applications of robotics. We have demonstrated the efficacy of this mapping using synthetic and laboratory generated image sequences.

Status of Project

The concept of ECLM was developed earlier and its role in segmentation of dynamic scenes [1] acquired using a moving observer was demonstrated. Later, applicability of this approach for motion stereo was shown considering synthetic and laboratory scenes [2]. We are now studying characteristics of the mapping and its effects on depth recovery.

We first studied the effect of mapping by studying how points from the cartesian space are mapped onto the CLM space. Our study [3] demonstrated the need for a better mapping technique. The earlier technique resulted in uneven mapping of near-FOE and peripheral areas of the image, making determination of image properties in the CLM space more error-prone.

Using the human eye as a model, a new interpolation technique for CLM was developed. Instead of using simple averaging in a 3x3 window, as done in our earlier work, a Gaussian weighted average in a window that varied with distance from the origin was used. We studied the performance of this interpolation scheme by using many synthetic and real images. A mapping to transform images back to the cartesian images from the CLM space was developed and used in our experiments. We transformed rotated and scaled versions of images and studied how they were affected due to the mapping. The new mapping method gave noticeably better results. The results of this were presented in a paper at the SPIE conference in April 1988 [4].

Feature matching is a necessary part of this stereo algorithm, as it is in binocular stereo. Points, lines, and regions are common image features used in matching. To get the best results with CLM, which of these features should be used and where should they be detected? We started studying the relative efficacy of points, lines, and regions in recovering depth using our approach. Experiments on synthetic images showed that all features could be used. For points, best results were obtained by doing the feature detection in cartesian space and then mapping the points. Depth determination using region matching works better if the regions are defined in CLM space. Lines gave ambiguous results.

Future Research

Error analysis of the method needs to be done, including sensitivity to the location of the origin of the mapping (the focus of expansion of the images). Methods for handling occlusion and for combining results over time need to be developed.

References

[1] Jain, R., "Segmentation of a frame sequence obtained by a moving observer," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, pp. 624-629, Sept. 1984.

[2] Jain, R., S. Bartlett, and N. O'Brien, "Motion Stereo Using Ego-Motion Complex Logarithmic Mapping", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, pp. 356-369, May 1987.

[3] Jain, R., S. Bartlett, and N. O'Brien, "Some Experiments in Ego-Motion Complex Logarithmic Mapping", in *Advances in Computer Vision and Image Processing*, Vol. 3, Ed. T.S. Huang, pp. 145-177, JAI Press, 1988.

[4] Bartlett, S. and R. Jain, "Motion stereo and ego-motion complex log mapping", Proc. SPIE Conf. on Digital and Optical Shape Representation and Pattern Recognition, Orlando, April 1988, in press.

2.1.3 Knowledge Based Computer Vision

Investigator: T. Weymouth

Problem Description & Research Objectives

The primary thrust of our research in knowledge-based vision is to develop methods of dealing with the uncertainty that arise in the various stages of scene interpretation. Scene data is presented as a stream of images; each image contains a noisy sample of the environment. To overcome that noise and reduce the data, abstractions of the image information are made. Further errors are introduced by this abstractions process, primarily from assumptions made in the processes. The effect of these errors can be overcome by using the redundancy in the scene and through the application of knowledge.

We are investigating the flexible use of two types of knowledge to reduce the uncertainty introduced by this abstraction process. The first type of knowledge is model-based descriptions of the environment. When the types of objects that are expected can be classified as to their geometric properties and can be described in a way useful to the data abstraction procedures, then this knowledge can be used to control the creation of appropriate descriptions of the scene. For example, if objects with straight-line edges are expected then the abstraction process for linking edges into straight lines and for finding sharp corners should be applied in the sequence of images arising from the scene.

The second type of knowledge is the information associated with object instances in the scene. As the description of the scene evolves it should be possible to apply rather specific abstraction processes. In a sequence of images, all from the same scene, following each other by short intervals, objects will not have moved that much. The recognition of object features (by abstraction) in one frame should be used to closely guide the recognition of those same features in subsequent frames.

A flexible control framework is needed so the selection of processes can be made effectively. In addition, during the process of experimental design and development, a flexible framework is desirable to accommodate the frequent changes of design typical of experimental development.

Approach

The three projects described below fit into a general architectural framework. In the framework, a computer vision system is assumed to consist of one or more cameras, each directly controlled by the computer, that deliver a stream of images to the system. The images are selectively processed to abstract features. These features may be grouped into further features through several levels of abstraction. Selected abstractions are integrated into a scene description, maintained by the system, that reflects the goal-specific interpretation of the scene.

The selection of features, the grouping of features, and the integration of those group into a description are controlled, in part by the past description of the scene. This is symbolic feedback. Current projects are concentrating on symbolic feedback as a means of reducing the error and uncertainty inherent in the abstraction processes.

Specifically:

- Geometric models can be used to control the types of edges that are abstracted from the image sequence. Further, as a scene description evolves, the current expectations of object position will lead to a prediction of where edges in the image should occur and exactly what type they will be. This allows limiting the bulk of the image processing to verification of expected feature values. We are experimenting with knowledge-guided interpretation of moving objects of known geometric characteristics.

- The computation of shape from optic flow has been an active and elusive goal. Our approach is to use symbolic feedback. In addition to investigating the constraints needed to derive shape from optic flow, we are investigating how predictions of shape can be used to facilitate the computation of optic flow.

- We are also investigating means of computing depth from stereo. Rather than use the pair of images from a single frame, however, we are developing methods for the integration of stereo disparity over several frames into an overall depth map. Assuming that the camera motion is known (e.g. under computer control) it is possible to predict where disparity points from one frame will lie in another. Using methods to reinforce consistent points and inhibit inconsistent ones, over time, we can build up a depth map. This project is new and has been given additional funding by General Dynamics.

To review: the three active projects are the knowledge-based interpretation of geometric forms, the computation of motion parameters from optic flow, and the computation of depth values from dynamic stereo. The knowledge-based camera control; the computation of motion parameters and the computation of depth values are mechanics that will be useful in any system for perception.

Status

The computational framework into which each of these projects is embedded is a blackboard architecture. We have implemented a vision workbench within CommonLisp (Lucid CommonLisp on the Apollos) which consists of standard graphics and display routines, image processing routines (such as the Canney edge operator and Burns straight line finder), and a blackboard architecture implementation, the GBB system from University of Massachusetts. To this software, we have added a digitizer and camera on an Apollo DN4000 workstation. Several experiments have been conducted within this vision workbench environment.

One of these experiments was a Blackboard based interpretation of two-dimensional polygons. As a demonstration project, we build a system which interprets noisy images of two-dimensional polygon forms using geometric models and knowledge guided feature grouping. The feature extraction and grouping routines are embedded in the blackboard framework. Straight line fragments extracted from images are grouped and long lines with adjacent corners form initial hypotheses. Three knowledge sources, two bottom-up and one top-down cooperated using a best-first strategy to advance those hypothesis which were both self-consistent and consistent with the object models through several levels of abstraction (corner/line, partial figure, full figure, scene). The final interpretation emerged as a singular, self-consistent, high-ranking, scene hypothesis. This project is being written up in a tech report.

Our earlier experiments in computation for motion parameters ran into problems when we began experimentation. Although the mathematical development was straightforward, the resulting process was numerically unstable. The results are critically dependent upon the computation of the image normal velocity and depth. We have now developed an algorithm for the computation of depth from image normal velocity and are working on a more robust algorithm for the computation of motion parameters from these quantities. These results are being written up.

Our most successful project has been in the computation of depth from stereo images. As a pair of cameras moves through the environment, when the camera motion is approximately known, the depth values from previous stereo computations can be used to reduce the computational load and uncertainty of the current stereo computation. To do this there needs to be a method of overcoming the uncertainty in camera motion. We have developed a method for registration of the camera, based on objects in the environment which allows the integration of depth value even when camera position is uncertain. A byproduct of this is that depth values are maintained in a camera-independent coordinate frame for subsequent interpolation into surfaces in the scene. Before publishing these methods we need to perform experiments to measure the accuracy and completeness of the depth information. Once these experiments have been performed, we will write a write a paper on this algorithm and the results.

Future Directions

Although funding within the project for these particular goals is ending, research will continue in the following directions:

- Interpretation of moving objects ultimately rests on an understanding of object geometry and geometric relations. We continue to investigate methods for applying geometric constraints.
- Understanding methods for the extraction of motion features is the bases for interpretation of motion images. We are working on algorithms for the extraction of motion parameters from optic flow and the segmentation of optic flow.
- The principle source of information for the reduction of error is the redundancy of nature. We will be working on methods for the integration of predictions (feedback) into the processing of incoming data. This research is in the area of active stereo vision.

2.2 Planning and Programming

2.2.1 Real-Time Collision-Free Robot Path Planning

Investigators: K. G. Shin and S. Jun

Problem Description and Research Objectives

One of the major problems with collision-free path planning for robots is the amount and complexity of computation required. The severity of this problem often leads to the design of unintelligent robots: they simply follow pre-planned paths. The usual problem associated with conventional methods lies in that all obstacles in the workspace are treated uniformly regardless of their size, shape, and orientation. To remedy this problem, we have proposed a unique measure that takes into consideration the size, shape, and orientation of obstacles in the workspace.

Almost all search algorithms use the *expand-and-select* paradigm to reach a goal node from a starting node. Depending on the search strategy used, it may or may not select one of the nodes that were expanded most recently. When the selected node is not one of the nodes generated most recently, the most recent node expansion may become useless. Also, in certain cases, the selected node may not have any successor, thus reaching a *deadend*.[1] When a deadend is encountered, the search will generate more nodes, and thus waste more

[1] A deadend is referred to as a node that does not have any children.

time as the number of deadends encountered in the search increases. To reduce the search time, the number of deadends encountered must be minimized.

The probability of meeting deadends depends on the search strategy used. That is, some search strategies tend to encounter deadends more often than others. On the other hand, there is a certain origin-destination pair that will always lead to one or more deadends regardless of the search strategy applied.

Approach

The workspace is divided into $l \times m \times n$ identical cells[2]. The object of a robot path planner is to find a sequence of neighboring free cells from the origin to the destination while minimizing a certain cost associated with the path.

The most commonly used cost is the length of thee robot path. In a Euclidean metric space E^d, the L_p-distance between two points, $x = (x_1, x_2, \ldots, x_d)$ and $y = (y_1, y_2, \ldots, y_d)$, is defined as:

$$d_p(x,y) = (\mid x_1 - y_1 \mid^p + \mid x_2 - y_2 \mid^p + \ldots + \mid x_d - y_d \mid^p)^{1/p}, \text{where } 1 \leq p < \infty$$
$$d_\infty(x,y) = \max(\mid x_1 - y_1 \mid, \mid x_2 - y_2 \mid, \ldots + \mid x_d - y_d \mid).$$

Though there exists an infinite number of L_p-metrics, only three of them have significance in robot path planning: L_1, L_2, and L_∞. The advantage of using L_2-metric over L_1- and L_∞-metric is its ability of describing the traversal distance. On the other hand, L_1- and L_∞-metric systems have the following advantages:

- The paths generated with L_1- and L_∞-metric are usually safer than those generated with L_2 -metric.
- The length of a path in a discrete space is the same as that of a path in a continuous space when L_1- or L_∞-metric is used.

Hence, we will limit our discussion to L_1-metric. That is, the cost of a path P, $C(P)$, is the L_1 length of P.

The lower bound of the probability that a cell meets a deadend can be obtained from the following procedure:

Procedure Lower_Bound
For every cube $v \in W$; W : The workspace.
if v *is free then*
begin

[2]The cells are squares in 2D and cubes in 3D.

initialize $I(w) \leftarrow 1$ for all $w \in W$; $I(w)$: Indicator function.

$I(v) \leftarrow 0$
for i=1 to maxdistance do
begin
Generated $D_i(v)$. ; $D_i(v)$: The set of cells whose distance from v is i.

for every $w \in D_i(v)$

$$I(w) \leftarrow \min_{u \in D_{i-1}(v) \bigcap D_1(u)} I(u)$$

end

$$p(v) \leftarrow \sum_{w \in W} I(w) g(v)$$

end
end{**Lower_Bound**}

The resulting probability partitions the workspace into several regions such that

$$u \in R(v) \Rightarrow S(u) = S(v) ,$$

where $R(v)$ is the region that contains the cell v and $S(v)$ is the set of cells that are not visible from v. In Fig. 4, any cell in R_1 is visible from any cell in the workspace except for those in R_9. Similarly, no cell in R_2 is visible from the cells in $R_8 \bigcup R_9 \bigcup R_{11}$, etc. Then, the probability that v meets a deadend becomes

$$P[C_v] = \sum_{u \in S(v)} g_v.$$

By partitioning the workspace as above, the problem can be divided into the following two subproblems:

Subproblem 1 (Inter-Region) *Find a sequence, P, of regions such that: $P = R_o R_1 R_2 \ldots R_k R_d$, where R_o is the region that contains the origin and R_d the region that contains the destination.*

Subproblem 2 (Intra-Region) *Find a method to traverse a region.*

The following lemma provides a useful property and proves that the solution of the subproblem **Intra-Region** can be obtained trivially.

Lemma 1 (Random-Path) *For any two cells, u and v where $u >_c v$, the shortest path between u and v can be obtained by a random sequence of available cells. Furthermore, such sequence always exists.*

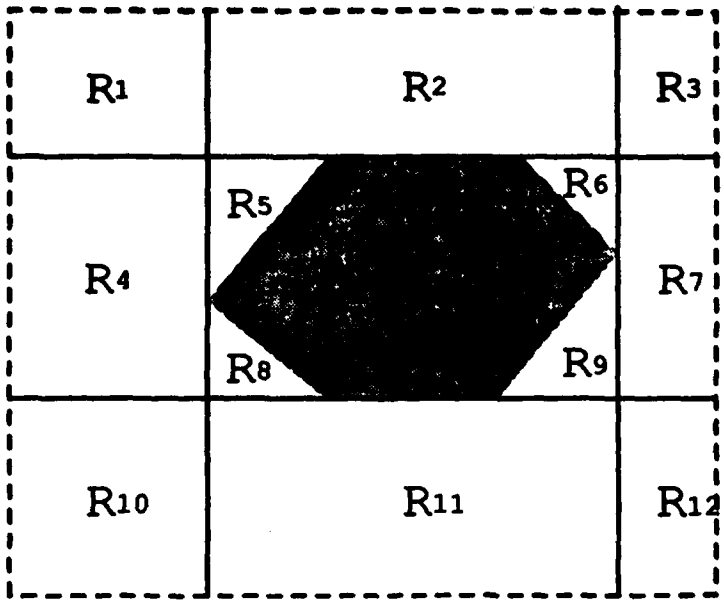


Figure 4: Partitioning of regions.

Status

We have shown that the problem of robot path planning can be divided into two subproblems, one of which has already solved. The solution to the other problem, **Iner-Region**, can be obtained by applying *Dynamic programming* or A^*. Since the number of regions is always smaller than the number of cells, the efficiency of these algorithms is greatly improved. Currently, we are implementing this partitioning for various search methods. Initial experiments have indicated that 3D path planning can be solved as simply as 2D problems.

Future Direction

We plan to expand our work in the following directions:

- Develop an algorithm, rather than using existing ones, that utilizes the specific nature of the partitioning of the workspace based on the probability of meeting deadends.
- Devise a representation model most suitable for this partition.
- Improve an algorithm that calculates the probability.

2.3 Automated Robot Task Planning

Investigator: K.B. Irani

Introduction

In the past year, research was directed towards developing a machine learning or automated knowledge acquisition approach for deriving search control knowledge for planning from problem solving histories in the form of problem solution traces. This goal has been attained. New concepts have been developed. Algorithms for learning has been designed. And a system, called GID3, has been successfully implemented and tested [1].

For the automation of the robot task planning, the efficiency of search to generate a good plan is the biggest issue. Since the planning problems are usually characterized by their huge search spaces, it is necessary for a planner to be equipped with sufficient control knowledge in the form of control decision rules or strategies to tightly constrain the search space and to effectively guide the search.

In the past, attempts have been made to mechanically deduce this type of knowledge from a problem formulation [2]. Although some good results are attained, it is found that the complexity of such approach would be formidably high for a realistic problem representation. However, such knowledge can often be brought out from traces of successful past problem solutions. This motivates us to turn to the machine learning approach which can be applied for automated knowledge acquisition.

Many approaches have been proposed before for automated knowledge acquisition. Among which the most appealing one is Quinlan's ID3 [3] which can generate a decision tree from a set of classification examples. Although this approach was originally designed for classification tasks, it can be easily adapted for learning control rules. However, this approach suffers from several drawbacks that impede its successful application in many domains. Our research in the past year identified causes for these problems and resulted in a novel approach which can avoid these problems without imposing any significant increase in computation.

Approaches Taken and Current Status

Approaches to automated knowledge acquisition range from the knowledge intensive explanation-based techniques to the purely syntactic methods for learning from examples. Our approach falls at the latter extreme. A learning system is first presented with a set of examples belonging to different concepts (classes). For each concept, the system is supposed to construct a description that covers all of its examples and excludes all the others. It is desired that such a system generate a minimal set of maximally general concept descriptions. This, in general, is an NP-problem, which makes a heuristic approach necessary. In the following, a currently popular classification learning system, ID3, is introduced first. Then the problems and weaknesses associated with ID3 are identified. Finally, our research work on overcoming these problems and developing a new learning system, GID3, is described.

ID3, developed by Quinlan [3], is a system which uses a hill-climbing search strategy with an efficient heuristic evaluation function. It uses the decision tree as the representation

for concepts, or in our terms, control rules. In order for the system to learn, a sequence of examples, called training examples, are fed into the system. These examples are represented as $m + 1$ tuples, with the first m elements being the values of m attributes of a physical object or phenomenon and the last element, the class of the example.

ID3 algorithm is recursively applied to every node of a decision tree. If a node contains a set of examples with more than one class, then the node is branched on if possible. An attribute is chosen such that its values can best discriminate the examples with different classes or at least will help most in reducing the ambiguity measured by the so called "ENTROPY" formula. This node is then branched on, each branching node carries with it a value of the selected attribute as a new condition and inherits from the parent node a subset of examples matching this condition. The program terminates when either every leaf node of the tree contains only examples of the same class, or all attribute values have been used in branching.

ID3 essentially employs a heuristic, hill-climbing, non-backtracking search through the space of possible decision trees. Although it is efficient, ID3 suffers from a weakness, namely, overspecialization, which causes it to often "miss" better decision trees for the same training data. The most pronounced problem leading to the weakness is the "irrelevant values problem." When ID3 chooses an attribute for branching on a node, it creates a branch for each value of that attribute that appears in the examples. The problem is that some of the values of that attribute may be relevant to the classification, yet the rest may not be. The subtrees generated by these irrelevant values result in overspecialized classification rules - rules that check for unnecessary or irrelevant preconditions. This problem motivated us to develop a learning mechanism which would avoid overspecialization during the generation of a decision tree.

Our new learning method is based on ID3, hence the same GID3 (Generalized ID3). The major change lies in the way of evaluating attributes and branching on any node of a decision tree. The evaluation now takes two steps. The first step is to evaluate all relevant attribute-values and the second step, to evaluate all modified attributes each of which contains only values surviving after the first test plus a "DEFAULT" value which groups all the values which fail the first test. Finally, an attribute is chosen whose modified set of values are used to branch on a decision tree mode.

In the first step, all the attribute-values appearing in the example set of a node are tested. For each attribute-values appearing in the example set of a node are tested. For each attribute-value pair $< A_i, V_j >$, a binary-valued "temporary attribute" denoted by "$< A_i = V_i >$ is created. This attribute takes only two values, TRUE and FALSE. When applied to the original set of examples, this attribute leads to a partition of examples into two sets, those satisfying "$A_i = V_j$" and those that do not. All these "attributes" are then evaluated using the entropy measure. The attribute with the least entropy, meaning most discriminative power,

is selected as the reference attribute. A user-determined tolerance value $TL(TL \geq 1.0)$ is then combined to form a "threshold" which is the product of TL and the minimum entropy value derived. This threshold is applied to all the attribute-value pairs so that only those whose measure are lower than the threshold are accepted for further test.

In the second step, a set of modified attributes are generated. An attribute is generated if it has at least one value surviving from the first test. The range of the attribute is modified to include all the surviving from the first test. The range of the attribute is modified to include all the surviving values plus a "DEFAULT" value. Therefore, the set of modified attributes is actually a subset of the original attributes and, the range of each modified attributes is smaller than the range of the corresponding original attribute. Now, all the modified attributes are evaluated using the ID3 heuristic measure and the best attribute is selected to perform actual branching on the decision node.

This new approach has two advantages over ID3. First, it effectively avoids branching on irrelevant values of attributes. Only the values that appear to be relevant, according to the information measure, are branched on. All other values are lumped together in one default branch. Second, it provides a system parameter, TL, which can be used to fine-tune the system's performance. TL specifies the degree of tolerance for deviation of the entropy measure of an attribute-value pair from the minimal entropy measure over all pairs. It is worth mentioning that TL provides a spectrum of system's learning behavior. At $TL = \inf$, the system behaves exactly like ID3, branching every time on all values of an attribute. The other extreme occurs at $TL = 1.0$, when only the attribute-value pairs whose entropy measure are minimum are branched on. This will, in general, result in a binary decision tree - branching on one attribute-value each item. As TL varies from 1.0 to inf, the system generates different decision trees. For a given set of data, we claim the existence of a setting for TL that results in the generation of a "better" decision tree than that of ID3. This claim has been empirically confirmed by our experiments.

Although it attains better decision trees, the new approach does not impose a significant increase in computational cost. The complexities of both algorithms are polynomial and of the same order. The extra computations involving TL are compensated for by the fact that the set of modified attributes are a subset of the original attributes, and that the ranges of these modified attributes are smaller than or equal to the ranges of the corresponding original attributes.

For our application, a postprocessing module is added to the learning method to transform a decision tree into a set of decision rules. Decision rules follow the format: "IF < *conditions* > THEN < *actions* >." Each rule actually corresponds to a path in the decision tree leading from the root node to a leaf node. The set of rules thus induced comprise a rule-base which can be used as a consistent set of search control strategies.

In order to compare the performance of GID3 with that of ID3, a series of tests were

performed along six dimensions, namely, (1) the number of decision nodes generated in a decision tree, (2) the percentage errors on classifying unseen examples, (3) the number of rules in the rule-base (or leaves in tree), (4) the total number of preconditions in all the rules, (5) the average example support per rule, and (6) the average number of decisions per example with respect to a set of test examples. Measure 1 tells the efficiency of the approach, measure 2 the accuracy of decisions generated, measure 3 the closeness to the ideal goal of finding a minimal set of rules, measure 4 the generality of the entire set of rules, measure 5 the applicability of the rules, and measure 6 the predicted efficiency of the generated set of decision rules.

Examples from several different domains have been used in testing. The results derived show that GID3 can outperform ID3 with all six performance measures described above. The decision trees generated by GID3 are more compact, more reliable, and more general than their ID3 counterparts.

References

[1] Cheng, J., Fayyad, U.M., Irani, K.B., Qian, Z., "Improved Decision Trees: A Generalized Version of ID3," *Proceedings of the 6th International Machine Learning Conference*, Ann Arbor, Michigan, June 1988.

[2] Irani, K.B., Cheng, J., "Subgoal Ordering and Goal Augmentation for Heuristic Problem Solving," *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, Milan, Italy, August 1987, pp. 1018-1024.

[3] Quinlan, J.R., "Induction of Decision Trees," *Machine Learning 1*, No. 1, pp. 81-806, Boston: Kluwer Academic Publishers, 1986.

2.4 Automatic Assembly Planning

Investigators: J. Wolter, A.C. Woo, R.A. Volz

Problem Definition

During the last three decades considerable advances have been made in the development of artificial intelligence systems to generate plans for building assemblies. These robot-planning systems have typically solved problems in which they are confronted with workspace containing several parts which must be rearranged to achieve a given goal position in a minimum number of steps.

While methods of solving this type of problem are of great interest, they are not generally very applicable to manufacturing problems. When designing an assembly line or workcell for the production of a new product, the engineer normally uses whatever fixtures and feeders are appropriate to the problem. Thus, the workspace geometry and the initial part placement will be outputs from the planning process rather than inputs.

To address this type of assembly problem, we are working on the design of a planner which takes as its input a geometric model of the assembly to be built, and produce the following:

1. a high-level description of the fixtures that will be needed.
2. for each fixture, a sequence of parts or subassemblies to be inserted.
3. for each part insertion, a trajectory that intersects no previously inserted parts.

The fixture specification would consist of a list of parts that the fixture must hold, and, for each, a description of the forces against which the part must be held. Each subassembly will be built in a fixture, and then used as a part in either another subassembly, or the final assembly.

For such a planner to be practical, careful consideration must be given to the criteria used to select plans. The criterion used in traditional robot planning is "minimum number of operations," but in assembly planning there are normally a very large number of plans with the same number of operations. Furthermore this fails to capture such important considerations as the difficulty of implementing the specified fixture and preforming the specified operations.

In this research project, we have been working to develop a practical approach to the automatic solution of assembly-planning problems with meaningful criteria.

Approach

Traditional robot planning systems usually pay very little attention to the problem of selecting trajectories with which to insert parts. Normally all parts are simply inserted from above. In assembly planning there is no predefined orientation in which the assembly will be built, thus "above" is not a meaningful concept. A more thorough treatment of trajectories is needed.

Until the plan is complete, we have no way of knowing which parts may already be in place when we try to insert a part. Rather than attempting to run a path planning algorithm for every possible combination of obstacles, we solve the problem in two stages:

1. Propose a list of likely insertion trajectories for each part.

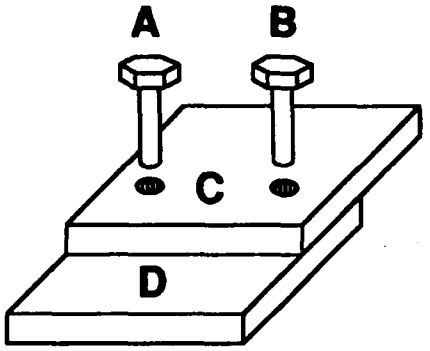


Figure 5: Sample Assembly

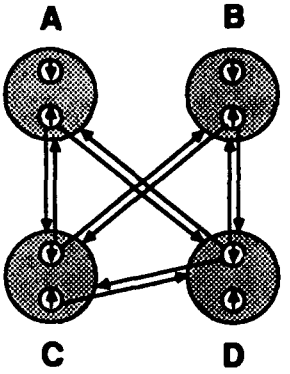


Figure 6: Assembly Constraint Graphs

2. Generate a plan using one trajectory selected from each part's list.

Insertion trajectories are proposed by recognizing certain common structures in assemblies. For example, when a series of parts is found on a shaft, trajectories parallel to the shaft will be proposed. When two parts are threaded together, a spiraling trajectory parallel to the screw shaft will be proposed. Vectors normal to contact surfaces are also likely candidates.

Once we have proposed an insertion trajectory for a part, we test it to see which other parts would block it if they were already in their final positions. The results of these test for the assembly shown in Figure 5 are diagramed in Figure 6. Here each part is represented by a large circle, and each trajectory which has been proposed for that part is represented by a small circle. For plate C there are two trajectories. It may be inserted either from above (assuming the orientation show in Figure 1) or from below. If we chose to use the trajectory from above, than both screw A and screw B must be inserted after C, so as not to block that trajectory. In Figure 2, this is represented by the arrows pointing from the down trajectory of node C to nodes A and B.

This type of diagram fully represents all geometric constraints in the assembly problem. It is the basic data structure to be used by our planner. Three operations can be used to develop this problem representation into an assembly plan:

1. Discard a possible trajectory from a part having more than one.

2. Add a constraint arrow from one part trajectory to another part.

3. Merge two or more parts into a subassembly.

If the result is a diagram in which each part has only one trajectory and all the parts in each subassembly are constrained into a strict linear sequence, then it is a valid assembly plan.

Several criteria have been developed to guide the application of these three operations to the constraint diagram. Our experimental system uses only the following three:

1. Directionality – As much as possible parts within a subassembly should be inserted from the same general direction. This tends to reduce both the complexity of the fixture and the number of degrees of freedom needed for the robot.
2. Manipulability – The more complex operations should be done with the smaller parts. That is, when screwing a bolt onto an automobile, we turn the bolt, not the automobile.
3. Fixture Complexity – The number of degrees of fixture which the fixture must hold things against should be minimized. It is easier hold a shaft and insert a series of parts onto it than it is to hold all the parts in line, and insert the shaft through them.

Several other criteria may be considered in greater detail later. These include, among others:

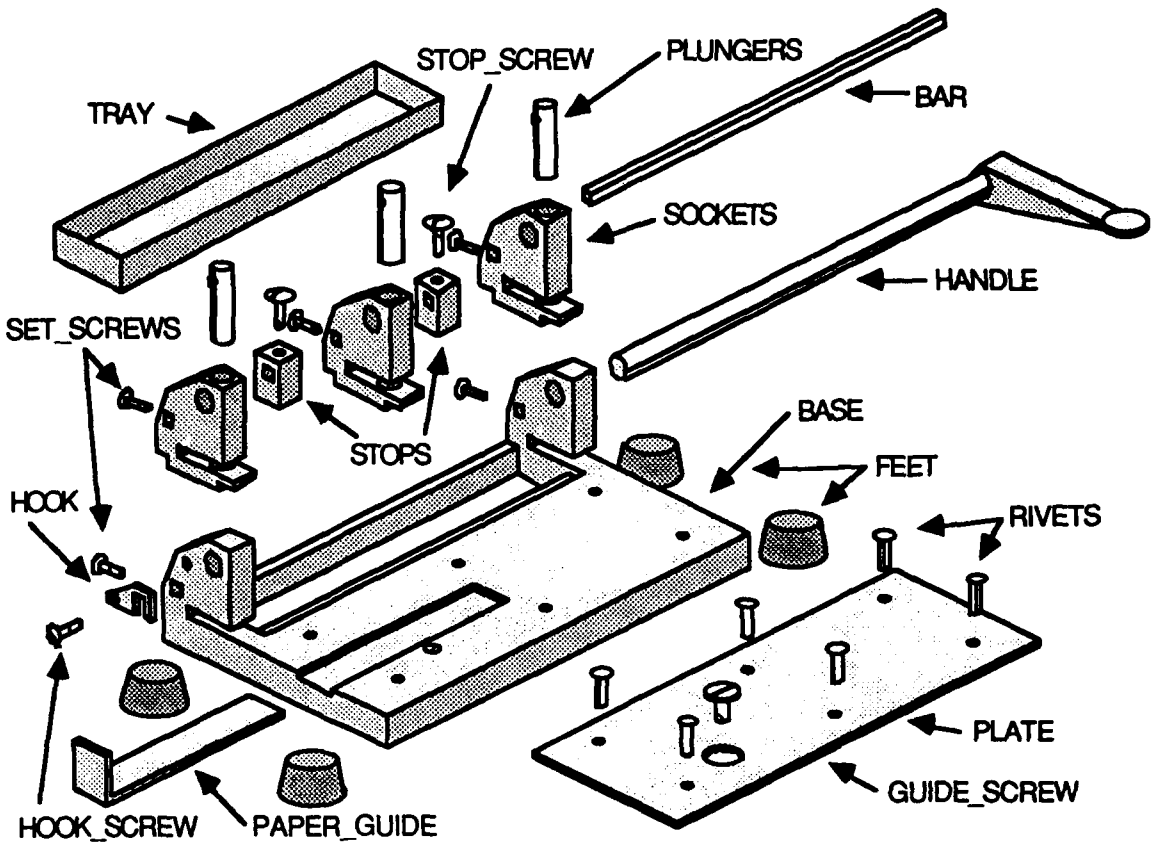
1. Uniformity – If there are similar parts or sets of parts, they should be built in similar ways. This reduces the number of different operations the robots must be able to perform. In some cases, it may be able to use the same fixture for several subassemblies.
2. Parallelism – When there are multiple robots, it may be advantageous if several operations can be preformed at once.
3. Locality – Operations which are preformed near each other should be done in sequence.

Clearly the relative importance of these different criteria would be different in different applications. In a single-robot workcell monodirectionality is more important than parallelism. On an assembly line, the contrary may be true. The relative weights of criteria would thus be supplied by the operator.

Status

In the last year, an experimental system, named XAP/1, has been implemented to solve assembly planning problems. XAP/1 is limited in that it does not handle subassemblies and has only three criteria installed, but due to the strongly modular design of the system, the installation of further criteria would not be difficult.

Though XAP/1 uses a very simple search technique, it's performance is quite good. Figure 7 shows a plan generated for the 34-part hole-punch assembly. Running on a Sun



(1) insert FOOT_B from above
(2) insert FOOT_D from above
(3) insert TRAY from above
(4) insert FOOT_A from above
(5) insert FOOT_C from above
(6) insert BASE from above
(7) insert STOP_A from above
(8) insert STOP_B from above
(9) insert SOCKET_C from above
(10) insert PLUNGER_C from above
(11) insert SOCKET_A from above
(12) insert PLUNGER_A from above
(13) insert SOCKET_B from above
(14) insert PLUNGER_B from above
(15) insert HANDLE from right
(16) insert HOOK from right
(17) screw HOOK_SCREW from left
(18) insert PLATE from above
(19) rivet in RIVET_D
(20) rivet in RIVET_F
(21) rivet in RIVET_B
(22) rivet in RIVET_C
(23) rivet in RIVET_E
(24) rivet in RIVET_A
(25) insert PAPER_GUIDE from left
(26) insert GUIDE_SCREW from above
(27) insert BAR from right
(28) screw STOP_SCREW_B from above
(29) screw SET_SCREW_C from behind
(30) screw SET_SCREW_E from behind
(31) screw SET_SCREW_A from behind
(32) screw STOP_SCREW_A from above
(33) screw SET_SCREW_B from behind
(34) screw SET_SCREW_D from behind

Figure 7: Plan Produced by XAP/1 for hole punch assembly

3/50 workstation, this plan, which is optimal for the set of criteria installed, required only $5\frac{1}{2}$ seconds of CPU time.

Future

Many avenues for further investigation have been opened by the XAP/1 system. We will briefly mention two here.

The primary shortcoming of XAP/1 is it's inability to handle subassemblies. To do this effectively, we believe a hierarchical approach to the problem would be suitable. This would involve adding a higher-level of abstraction on top of the current low-level view of the assembly. At this higher level, subassembly decisions could be more easily made.

We would also be interested in studying methods of implementing this type of planner on a parallel processor. XAP/1 is designed in a strongly modular fashion. It would be possible to have different branches of the search run on different processors, or to run different criteria modules on different processors.

2.4.1 Research Report on Part-mating Planning in the Presence of Uncertainty

Investigator: R.A. Volz, J. Xiao

Problem Description and Research Objectives

The manual development of robot programs for assembly is a laborious, error prone task. It would be highly desirable to be able to derive them automatically from design information. In order to do so, several important problems must be solved: 1) automatic determination of an assembly plan (sequence of assembly operations), 2) automatic determination of parts presentation and fixturing, 3) gross motion planning, 4) fine motion planning, and 5) automatic generation of error recovery routines[6]. This report addresses the last of these problems.

It has often been stated that the largest part of any robot assembly program is made up of fix ups to handle things that don't quite work as planned. Algorithms for generating these fix ups are crucial to successful automatic program generation. Problems in the execution of nominal robot programs arise for the following reasons:

- *mechanical* and *control errors*: robots and all other mechanical devices are only accurate to within certain bounds;
- *sensor errors*: all sensors have limited sensitivity and accuracy;
- *manufacturing tolerances*: different instances of the same model are not exactly identical because of manufacturing limitations.

Often such errors can lead to the failure of a nominal program that would theoretically accomplish a task such as part mating. The automatic generation of robot programs for assembly cannot be successful until some means of either avoiding or correcting these errors is found.

Several approaches towards handling these uncertainties have appeared in the literature. Taylor[11] introduced an error-propagation method to estimate compound errors from the uncertainty bound or tolerance of each individual part involved in a task (including the robot hand). Brooks[1] extended Taylor's method by making the process backward, so that the constraints on some compound errors to make the task succeed can lead to constraints on individual parts. This method, however, suffers from high computational complexity. And it does not provide means to reduce errors dynamically. The inductive learning approach by Dufay and Latombe[3] corrects run-time errors by adding rules into the system as a corrective plan. In this approach, error-handling is not fully automatic, since rules must be provided by human users.

The pre-image approach introduced by Lozano-Pérez, Mason, and Taylor[5], and simplified by Erdmann [4] incorporates the effect of uncertainty directly within one planning phase. The goal is to create motion plans that will avoid errors. However, this approach has unsolved theoretical problems and its applicability to practical cases is questionable. A more practical force control method was developed by Whitney[13]. His remote center compliance device can correct small insertion errors for a peg-in-hole task by applying correct forces to the peg, but it only works when the peg is in or partly in the hole.

All the previous approaches contribute to solve the problem in some ways, but none fully solves it. The problem is so complex that new solutions are still in need.

Approach

The approach we present is based upon three hypotheses:

1. The problem cannot be solved in general. Design and motion constraints relating nominal design parameters, tolerances and sensor error parameters must be derived concurrently with the planning algorithms, if success is to be guaranteed.

2. A two phase planning process, an off-line nominal planner and an on-line replanner to correct for run-time errors, simplifies the overall system.

3. Replanning (when failures occur) must involve knowledge about the contacts between the part being moved by the robot and those of other parts involved in the process.

We are to develop algorithms for robotic assembly based upon these principles and the use of *imperfect* position, force, moment and contact sensors in the presence of control errors.

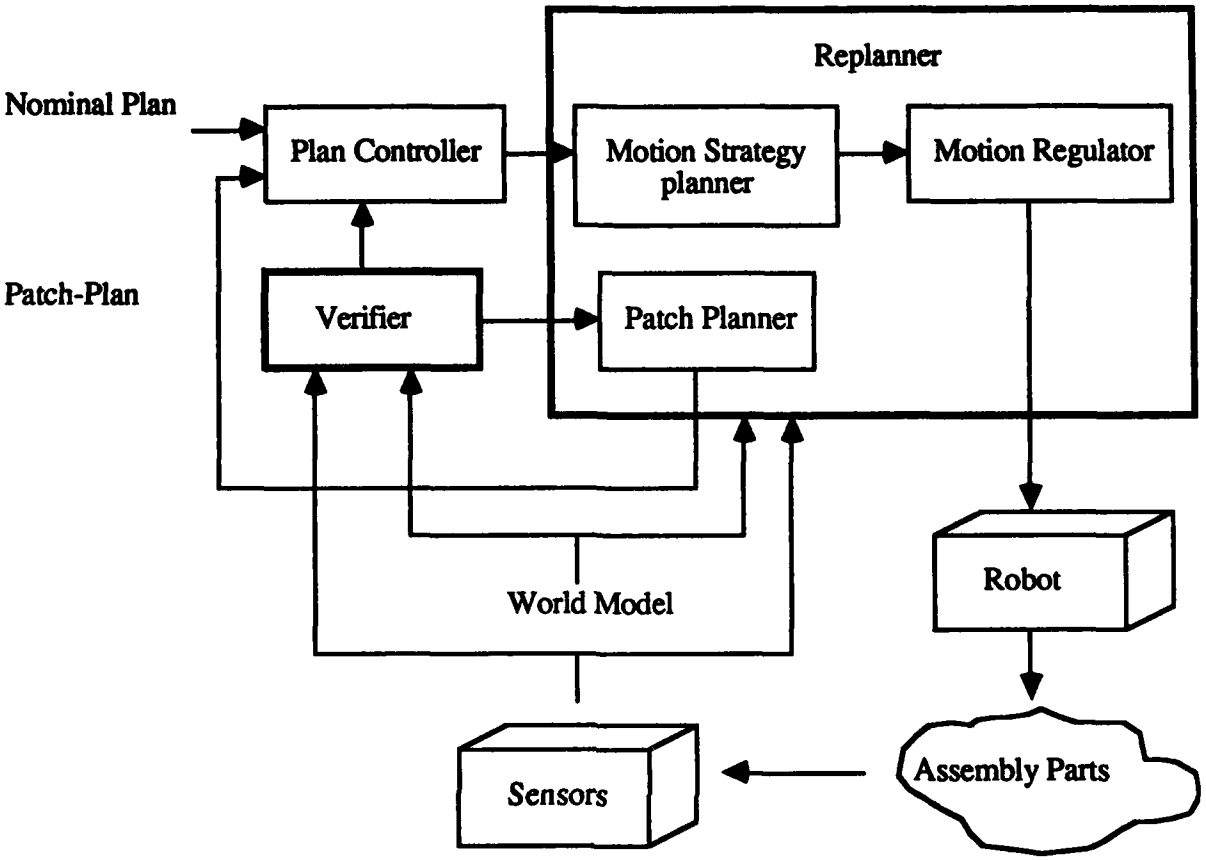


Figure 8: System framework

Further, we seek algorithms whose success we can assure in spite of certain classes of sensor, control and manufacturing imperfections.

The system framework relevant to the proposed approach is shown in Fig. 8. The system consists of three parts: a verifier, a replanner (consisting of a patch-planner, motion strategy planner, and a motion regulator), and a plan controller. Given the task environment(i.e. the robot, the assembly parts, and the sensors), a nominal plan, and a world model which consists of geometrical and physical descriptions of the task environment, the verifier identifies contacts among the assembly parts whenever an unexpected collision occurs[3]. Then, the patch-planner generates patch-plans to resolve the collision. Each patch-plan generates a path connecting the unexpected configuration to one from which the nominal control can continue. The motion strategy planner selects specific control algorithms for individual plan segments taking into account the system uncertainties, and the motion regulator executes

[3]There are presumed to be force/moment guards that stop motions which make the unintended move.

both the nominal and patch-plan motions utilizing imperfect sensor readings as required by the individual control strategies. The plan controller plugs in a patch-plan to the nominal plan whenever an unexpected contact is verified. It acts like a switch and is controlled by the verifier.

Fundamental to our (replanning) approach is the exploration of relationships between tolerances for successfully accomplishing a task and bounds on uncertainties, leading to the development of proper design constraints on the system uncertainties, which, if satisfied, can guarantee the replanning to be successful.

Status

The major issues involved in our approach are

1. how to describe and classify contacts among assembly parts,
2. how to identify or verify a contact,
3. how to describe or model the sensor uncertainties, control and mechanical errors, and geometric tolerances,
4. how to generate patch-plans based upon knowledge of contact and other sensory information,
5. how to obtain a (replanning) motion strategy as well as necessary design and motion constraints to guarantee the success of the strategy,

of which, the first three issues and the last issue have been largely resolved in our previous reports. We will briefly review those issues before we introduce our solution to the fourth issue below.

We have introduced the notion of *contact formations* as a set of *elemental contacts* (Fig. 9)[4] among the objects involved [12][2], to describe different topological contacts among the parts being assembled. We have also described a technique for, under certain design constraints, uniquely determining the contact formation in which the parts lie in spite of sensor and control errors[12][2].

We have supposed that only position/orientation and force/torque sensors are available in the system. Thus, four kinds of errors have been modelled: geometric uncertainty, position/orientation sensor uncertainty ϵ_p and ϵ_o, force/moment sensor uncertainty ϵ_f and ϵ_m, and robot motion uncertainty ϵ_v, ϵ_ω, v_r, and ω_t[12].

Nevins and Whitney[9] have determined that there are 12 principal assembly operations(Fig. 10), of which most common assembly tasks can be composed. The study[9] also

[4]An elemental contact is a contact between exactly two topological elements.

1. Surface-Surface

2. Surface-Edge

3. Surface-Vertex

4. Edge-Edge

5. Edge-Vertex

6. Vertex-Vertex

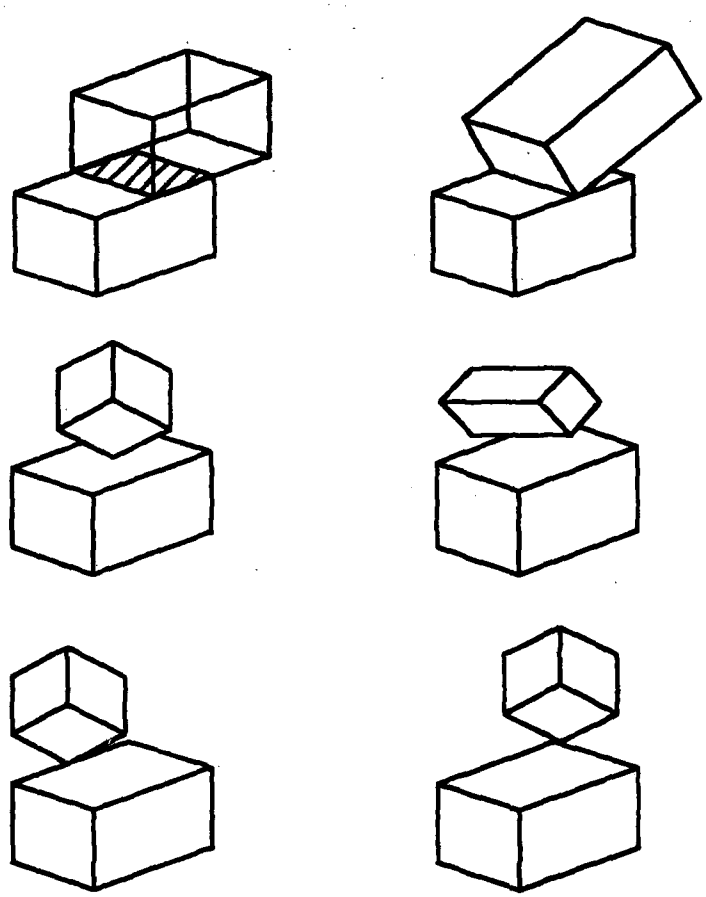
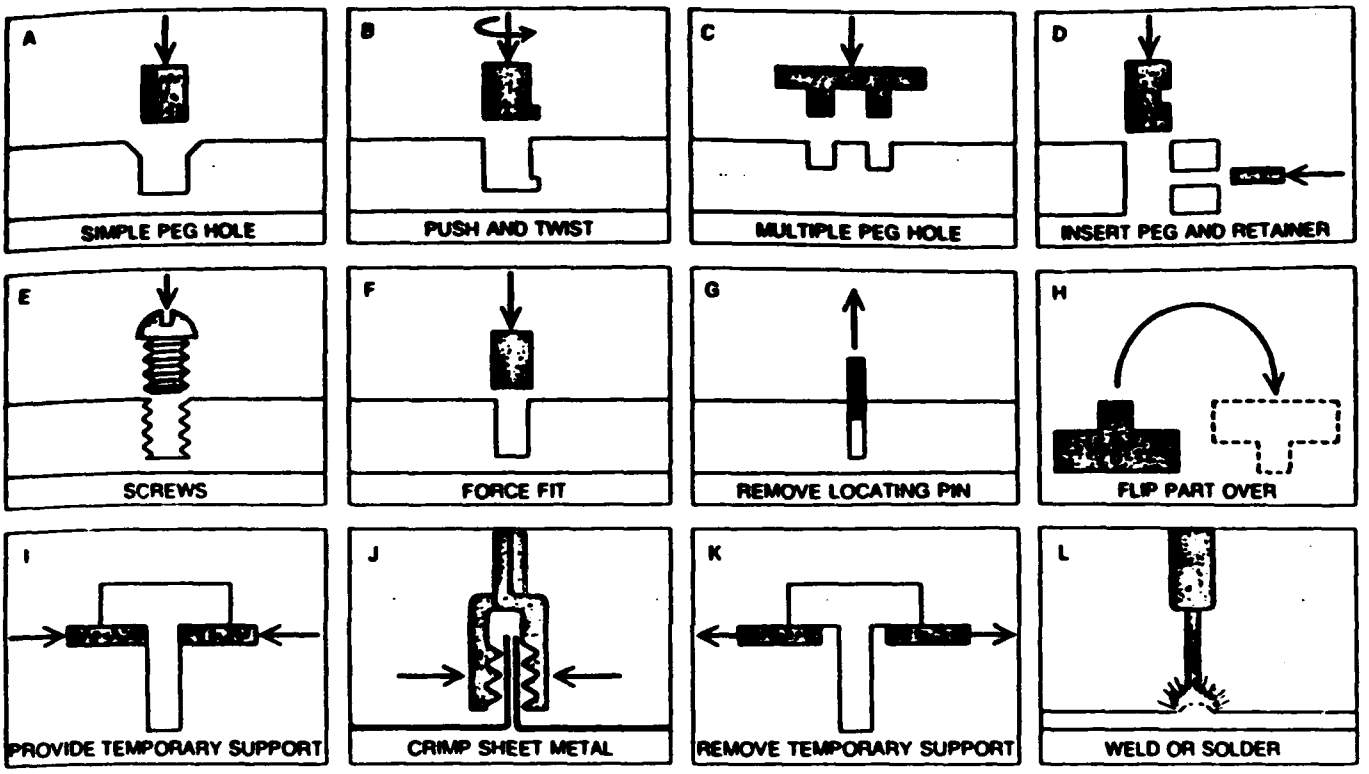


Figure 9: Elemental Contacts



TYPICAL MANUFACTURING TASKS were identified by taking apart and reassembling a variety of products and their components, including a refrigerator compressor, an electric jigsaw, an induction electric motor, a toaster-oven, a bicycle brake and the automobile alternator used in robot assembly project. All the items could be assembled with various combinations of 12 operations depicted.

Figure 10: Typical assembly tasks

reveals that among the 12 operations, those involving insertion are the most difficult and error-sensitive tasks. We have thus focused replanning on insertion, or peg-in-hole tasks.

For a peg-in-hole task, it often occurs that the peg is not put properly above the hole before the insertion, so that the peg hits the entry surface of the hole and the task fails. Although the phenomenon is most commonly studied in the area of uncertainty handling, previous research either does not guarantee the success of the task or is too complex to be practical. To handle the problem, we have developed a simple replanning motion strategy for peg-in-hole tasks, with proper design and motion constraints sufficient to the success of the strategy in two cases[14]:

1. using position/orientation sensing only,

2. using force/torque sensing to compensate position/orientation sensing.

The objective was to move the peg to a point properly aligned above the hole, with error within the tolerance, so that the insertion can be done. We have tested the constraints and the motion strategy by simulation, which showed that the constraints were correct and practically reasonable, and could be relaxed somewhat in real cases, with excellent results still obtained[14].

The fourth issue (i.e., how to generate patch-plans) is the major concern of this report. The patch-planning phase is entered by the premature termination of a commanded motion by one of the force/moment guards, resulting in a contact formation between the part being moved and the environment other than that planned[5]. The patch-planner then generates a patch-plan to resolve the unexpected contact formation. Each patch-plan generates a path connecting the unexpected configuration to one from which the nominal control can continue.

If we assume polyhedral objects only, then for each elemental contact between two objects, a contact plane can be determined[6]. Since a patch-motion starts from a contact formation in which a nominal motion terminates , the patch-motion can, in many cases, be guided by the contact planes of the elemental contacts involved in the contact formation, i.e., compliantly sliding or rotating the peg being moved along contact planes[7][8][10]. In this way, the dimensionality and the associated uncertainties of the motion can be reduced. Since the actual path taken by the robot will generally be close to the nominal, a simple sliding motion (of the peg) along one or more contact planes can often be sufficient. In the rest of the report, we present a strategy to generate patch-motions for failures of peg-in-hole tasks(Fig. 11), which mostly uses simple compliant motions.

[5]We view the goal state of a motion command as consisting of a set (often a singleton) of contact formations and a set of position and force constraints.

[6]We do not consider vertex-edge, vertex-vertex contacts, and edge-edge contacts with two edges fully contacting each other, since those cases almost do not exist *alone* in reality.

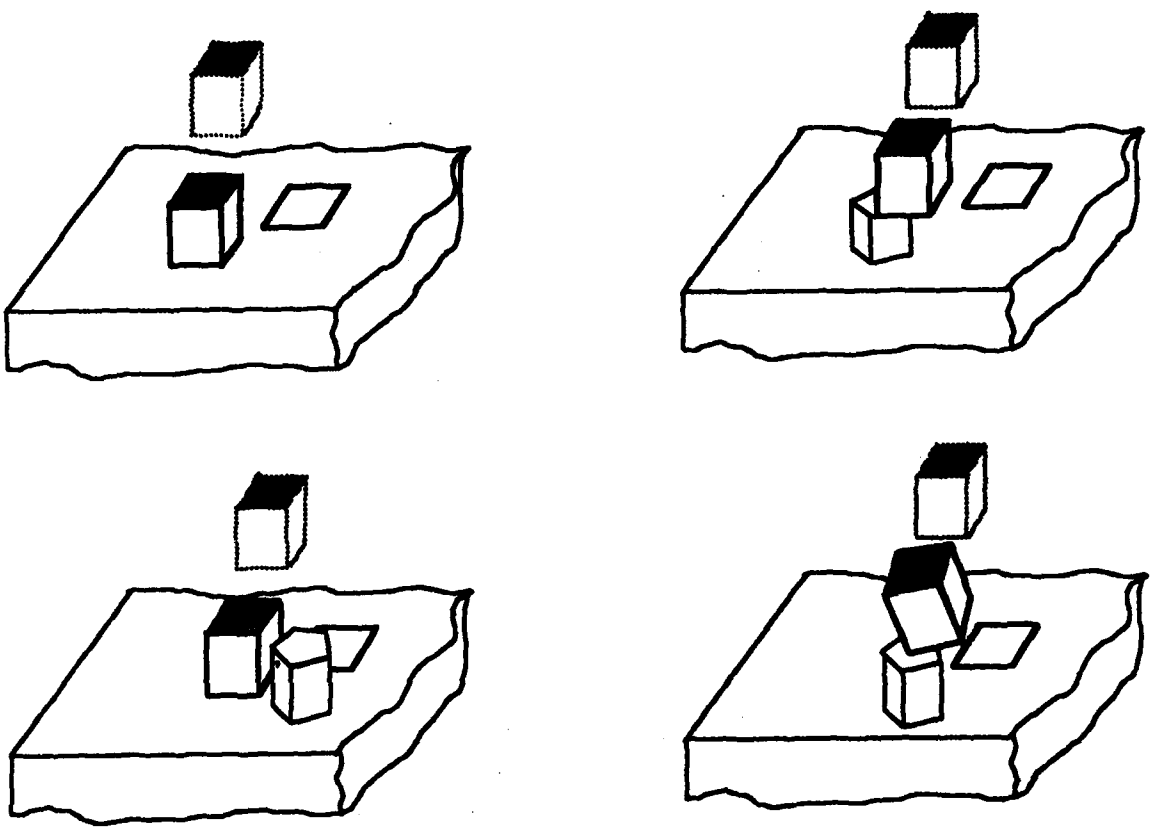


Figure 11: Different failures of a peg-in-hole task

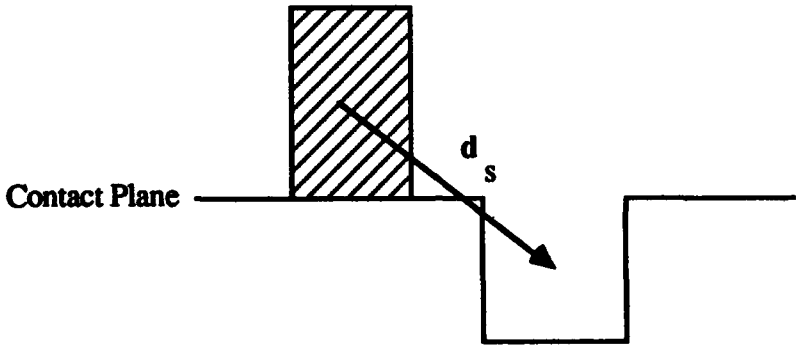


Figure 12: Contact blocks the ideal (shortest) path

We have developed a simple contact formation based patch-planning strategy (for patch-plan generation). Ideally, we want the patch motion following a straight-line path determined by the (sensed) positions of the peg ($\mathbf{P}_s^p$) and the hole ($\mathbf{P}_s^h$) or determined by moment sensing[14], denoted as $\mathbf{d}_s$. However, in many cases it won't work, simply because the current contact blocks the way (Fig. 12). We thus use the projection of the straight-line path $\mathbf{d}_s$ onto the contact plane(s) to obtain a patch plan involving compliant motions. The patch-plan path obtained consists of two or more motion segments, distinguished by the number of contact planes involved (or the degrees of freedom constrained). There are cases, however, when some elemental contacts do not block $\mathbf{d}_s$(Fig. 13), then projections of $\mathbf{d}_s$ on contact planes determined by those contacts are not necessary. Therefore, before the projections can be done, a test is needed on whether an elemental contact, which is involved in the contact formation the peg lies, blocks the ideal path $\mathbf{d}_s$. In general, a dynamic and sequential way of generating patch-plans can be as follows:

1. verify the current contact formation and obtain the (sensed) location of the held peg, if the peg has reached one of the goal contact formations, go to Step 10;
2. if the (sensed) orientation of the peg is not within proper bounds, (compliantly) rotate the peg to a proper orientation (guided by the orientation sensor), then verify the (new) contact formation again, which mostly involves a face-face contact between the peg and the environment;
3. number contact planes C_1, C_2, ..., C_n;

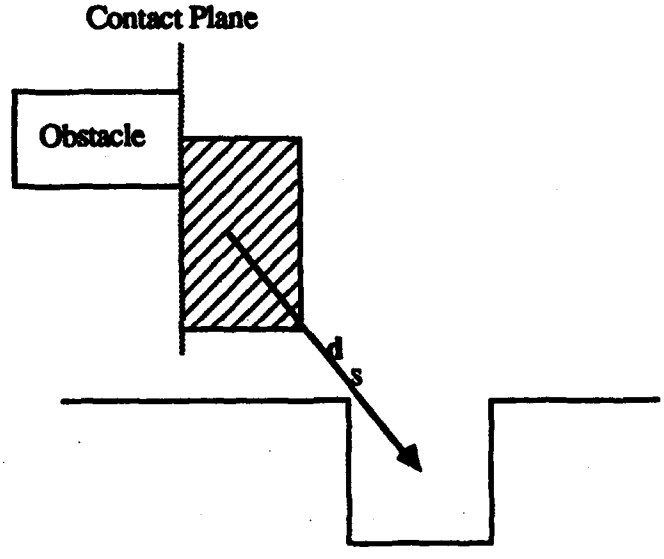


Figure 13: Contacts that do not block the ideal path

4. $i <= 1$;

5. if C_i blocks $\mathbf{d}_s$, go to Step 8;

6. $i <= i + 1$;

7. if $i < n + 1$, then go to Step 5, else go to Step 20;

8. project path $\mathbf{d}_s$ onto C_i, and obtain path segment $\mathbf{d}_{ci}$[7];

9. if $i = n$ then go to Step 13;

10. $j <= i + 1$;

11. if C_j blocks $\mathbf{d}_{ci}$ then project $\mathbf{d}_{ci}$ onto C_j to obtain $\mathbf{d}_{cj}$, and $i <= j$;

12. if $j < n$ then $j <= j + 1$ and go to Step 11;

13. $j <= 1$;

14. if C_j blocks $\mathbf{d}_{ci}$, go to Step 17;

15. $j <= j + 1$;

16. if $j < n + 1$ then go to Step 14, else go to Step 18;

17. $\mathbf{d}_{ci} <= -\mathbf{d}_{ci}$;

18. move the peg compliantly under the guidance of $\mathbf{d}_{ci}$ until the motion is stopped by force/torque (sensor) guards (where if there is no new collision encountered, the contact formation reached should involve less elemental contacts or even no contacts[8]);

19. if the contact formation is not null, go to Step 1, otherwise, obtain the new $\mathbf{d}_s$ formed by the current (sensed) positions of the peg and the hole;

20. move the peg along $\mathbf{d}_s$ until the force/torque (sensor) guards stop the motion, then go to Step 1;

21. end.

The ideas behind the general strategy are:

1. to keep the peg in a proper orientation as much as possible,

[7]If the projection is zero, pick up a unit vector on C_i to be $\mathbf{d}_{ci}$.

[8]We view the case where the held peg has no contact with the environment as in a null contact formation.

2. to keep the patch-plan path as short as possible,

3. to reduce, as much as possible, the dimensionality of patch-planning motions, in other words, to make patch-motions as constrained as possible in order to minimize the effects of uncertainties.

It can be shown that the strategy can always succeed if the polyhedral obstacles around the hole are convex and that the minimum distance between two obstacles is greater than the width of the peg.

Future Direction

We will continue our research on devising a generalized replanning strategy (within the system framework shown in Fig. 8), to handle failures and uncertainties occurred in assembly using robots. Along with the theoretical esearch, we will experimentally test our results on the puma 560 robot upon necessity, to further explore the applicability of our approach.

References

[1] R. A. Brooks, "Symbolic error analysis and robot planning," *Int. J. Robotics Res.*, 1(4):29, 1982.

[2] R. Desai, "On fine motion in mechanical assembly in presence of uncertainty," *Ph.D. Dissertation*, Department of Mechanical Engineering, the University of Michigan, December 1987.

[3] B. Dufay, and J. C. Latombe, "An approach to automatic robot programming based on inductive learning," August 1983.

[4] M. Erdmann, "Using backprojections for fine motion planning with uncertainty," *Int. J. Robotics Res.*, 5(1):19, 1986.

[5] T. Lozano-Pérez, M. T. Mason, and R. H. Taylor, "Automatic synthesis of fine-motion strategies for robot," *Int. J. Robotics Res.*, 3(1):3, 1984.

[6] T. Lozano-Pérez, 1983. "Task planning," *Robot Motion: Planning and Control*, M.Brady et al. Eds.

[7] M. T. Mason, "Compliance and force control for computer controlled manipulators," *Robot Motion: Planning and Control*, M.Brady et al. Eds., 1981.

[8] M. T. Mason, "Compliant motion," *Robot Motion: Planning and Control*, M.Brady et al. Eds., 1982.

[9] J. L. Nevins, and D. E. Whitney, "Computer-controlled Assembly," *Scientific American*, 238(2):62, February 1978.

[10] M. H. Raibert, and J. J. Graig, "Hybrid position/force control of manipulators," *Robot Motion: Planning and Control*, M.Brady et al. Eds., 1981.

[11] R. H. Taylor, "The synthesis of manipulator control programs from task-level specifications," *AIM-282*, Stanford Univ., 1976.

[12] R. Volz, J. Xiao, and R. Desai, "Contact formations and design constraints: A new basis for the automatic generation of robot programs," *NATO ARW: CAD Based Programming for Sensor Based Robots*, July 1988.

[13] D. E. Whitney, "Force feedback control of manipulator fine motions," *J. Dynamics Sys. Measurement, and Control*, 98:91, 1977.

[14] J. Xiao, and R. Volz, "Design and motion constraints of part-mating planning in the presence of uncertainties," *Proc. IEEE Int. Conf. Robotics and Automation*, April 1988.

2.4.2 Automatic Determination of Retrieving Paths in a Semantic Data Model

Investigators: Y.C. Lee, C.H. Cho, E. Armstrong

Problem Descriptions and Research Objectives

Semantic models for databases [2, 5] that result in more meaningful databases and provide meta-knowledge about the data have been considered good candidates for the integration of various database management systems. For example, any conceptual schema proposed for the ongoing PDES (Product Data Exchange Specification) project has been required to be presented as an integrated semantic data model using the IDEF1X modeling technique developed by the U. S. Air Force. However, methods to exploit this increased information so that the task of the user is simplified have been few in comparison to the number of semantic models in recent years. Accordingly, semantic data models, albeit highly expressive, also present the user a much more complicated view of data to deal with.

In order to be more expressive, more evolvable, and more disciplined, the semantic data model uses a number of high-level constructs to capture the semantics. Nodes in the model are classified into four types: *interaction*, *generalization*, *aggregation*, and *member*. Each *interaction* node combines two participating nodes to represent the relationship between them. Each *generalization* node combines a number of individual nodes, each being a special subtype of the generalization node. Each *aggregation* node groups together several nodes and associates them as its properties. Each *member* node stands for a property which can be printed or compared to a constant value.

As more semantic nodes are defined, the model requires more relations in the relation scheme through which a user specifies queries. Consequently, a seemingly natural and concise query would require a very complicated formulation for the query to be evaluated. The reason is that, in formulating a query, the user is in fact navigating the model to identify all relations that are involved and the associations between them. Although most relational query languages are nonprocedural (i.e., it is not required to specify *how* the data would be retrieved), it is still needed for the user to inform the system of *where* the data resides and *what* interrelations exist between data items. As the number of relations increases, query specifications become lengthy and complicated. Examples that illustrate these problems can be found in the progress report of last year[1].

So, queries that make sense to the users do not necessarily present complete formulations to the relational query language processor. A question can however be raised:"Should users deal directly with the relations which implement the semantic data model?" If the answer is no, then "how can user's queries be evaluated directly?" The objective of this research is therefore to equip the semantic data model with inferential capabilities so that users need not to reformulate queries into more complicated but executable forms. More specifically, the requirement of *where* specification can be eliminated and the *what* specification be simplified. We believe that, with the meta-knowledge available through data semantics, an intelligent query processor would be able to directly evaluate user's queries.

Approach

In general, each algebraic operation performed on the database yields some derived data which is used in another operation with some other data set from the database to obtain data which is in a sense closer to the data required by the query. Thus, from the starting point till the final stage when the answer is available, a number of relations need to be examined in a specific order and manipulated. The collection of relations can be referred to as a path. The process of automatically finding the path and answering the query is called the path determination problem.

Our first approach tries to find the complete path from the graph which represents the semantic data model[3]. Since then, a number of modifications have been made to reduce the cost required to find the path and to evaluate the query. To define all possible types of queries on a semantic data model, we have earlier proposed an enhanced but simple query language named ESQL[4]. The design of the syntax is based on the notion of the expressions *object.property* and *event(object,object)*. The object can be the name of a node of any type except the member type. The property can be the name of a node of any type that seems to be a property possessed by the prefixing object. The event can be the name of an interaction node or a generalization node with interaction nodes as subtypes. The event is participated by the two specified objects. This notion, if supported, would relieve the user of the burden in identifying necessary associations within each generalization hierarchy and thus simplifies

greatly the query formulation.

The ESQL language processor is divided into two parts. Without referring to the semantic data model, Part I parses the input query and constructs a query tree. Each nonleaf node of the query tree corresponds to a relational operation that either compares two expressions or combines (*AND/OR*) two predicates of the *WHERE* clause. The leaf nodes are either *object.property* or *event(object,object)*. As mentioned in previous progress reports, due to property inheritance, relationship inheritance, and property cascading, the content of each leaf node is not readily available from a single relation. Obviously, it is Part II that consults with the semantic model to locate the paths which are required to evaluate such expressions. In the following, we will show an example and the key algorithms.

Examples

In Figure 14, a pseudo semantic data model is shown, which consists of four generalization hierarchies, each being distinguished by the first subscript character, namely, a, b, c, and d. Nodes labeled by A, G, I, and M stand for Aggregation, Generalization, Interaction, and Member, respectively. Hierarchies c and d include the various interactions between hierarchies a and b. The reason why a pseudo model is used here is to include a larger variety of paths so that the path finding approach and its supporting algorithms can be better illustrated.

ESQL has a very simple and natural syntax. A user needs to know only the following about a specific model: (1) the information available within each hierarchy, namely, the name of each node, be it an object, an event, or a property; and (2) the interactions between two hierarchies, namely, the name of each interaction node. For instance, he/she can easily construct the ESQL query below to answer the question "Find the property M_{d_1} of the event A_{d_3} between objects A_{a_4} and A_{b_1} where the property M_{a_2} of the object A_{a_4} equals to a constant *aaa*, the property M_{c_1} of the event A_{c_1} between objects $A_{a_{10}}$ and A_{b_1} is the same as that between objects A_{a_4} and A_{b_3}, the property $M_{a_{14}}$ of object $A_{a_{10}}$ is equal to that of $A_{a_{11}}$, and the property $M_{a_{14}}$ of object $A_{a_{11}}$ is equal to the constant *bbb* ?"

FIND	$A_{d_3}(A_{a_4}, A_{b_1}).M_{d_1}$
WHERE	$A_{a_4}.M_{a_2} =$ *"aaa"*
AND	$A_{c_1}(A_{a_{10}}, A_{b_1}).M_{c_1} = A_{c_1}(A_{a_4}, A_{b_3}).M_{c_1}$
AND	$A_{a_{10}}.M_{a_{14}} = A_{a_{11}}.M_{a_{14}}$
AND	$A_{a_{11}}.M_{a_{14}} =$ *"bbb"*

In specifying such a query, the user is relieved of details such as: (1) how the underlying relations are defined; (2) how and what properties a composite property is decomposed into; and (3) how each hierarchy is structured, namely, what node is directly superior or

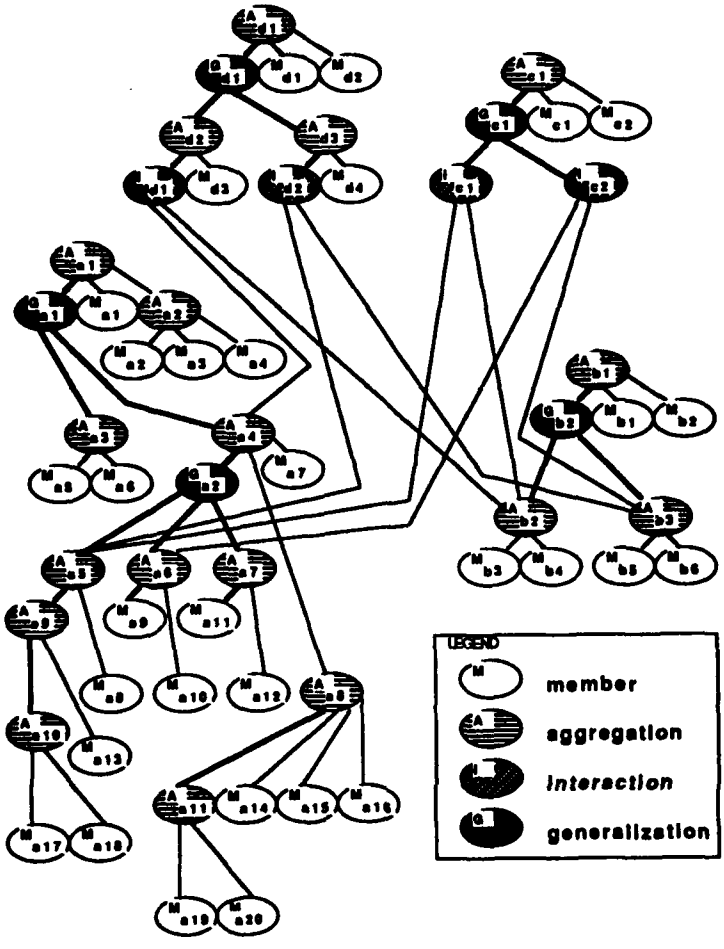


Figure 14: A Pseudo Semantic Data Model.

subordinate to a node. In a sense, the user is not required to know how the information is structured, as long as he/she is roughly aware of the information available.

Part I - Logical Path

As indicated earlier, Part I of the ESQL processor parses the query and constructs a query tree. The query tree indicates how the final result will be computed once each leaf node is evaluated. Since it does not correspond to any path embedded in the semantic graph, it is called the *logical path.*

While the evaluation of the nonleaf nodes in the query tree can be handled as usual, the leaf nodes that distinguish ESQL from SQL require the finding of retrieving paths. In constructing the query tree for the above example, eight leaf nodes are identified, one for

each expression below:

(1) $A_{d_3}.M_{d_1}$
(2) $A_{d_3}(A_{a_4}, A_{b_1})$
(3) $A_{a_4}.M_{a_2}$
(4) $A_{c_1}.M_{c_1}$
(5) $A_{c_1}(A_{a_{10}}, A_{b_1})$
(6) $A_{c_1}(A_{a_4}, A_{b_3})$
(7) $A_{a_{10}}.M_{a_{14}}$
(8) $A_{a_{11}}.M_{a_{14}}$

In contrast to logical path, each expression above is associated with one or more physical path, i.e. chain of nodes in the semantic graph, which will be determined by Part II. Along the path, related data can be collected to eventually evaluate the underlying ESQL expression. How each path is found and how its corresponding expression is translated into a relational algebraic expression will be explained in the following.

Part II - Physical Path

In implementing the semantic data model by relations, each node except those of member type, is augmented with a column of surrogates. A *surrogate* is a permanent system assigned pseudo-attribute that functions as a key for the node. The added advantage of having surrogates for each node is that the inclusion of the node into its parent node can be done by including the surrogate as an attribute in the parent node with the attribute name being the name of the node. Let's denote the relation associated with each node, except member type, by r_x where x is the name of the node. The surrogate column for node x is named as attribute $x\#$.

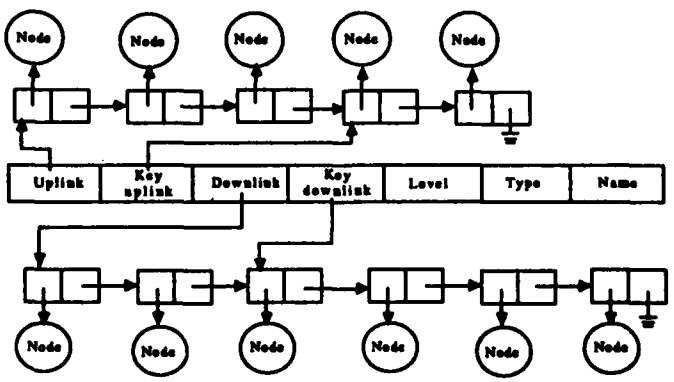


Figure 15: Node Declaration of the Semantic Graph.

To facilitate the path finding, the semantic data model is represented as a graph in which each node is defined as in Figure 15.

The field *Uplink* stores a pointer to the linked list consisting of all immediate superior nodes except *interaction* type. The field *Key-uplink* points to the linked list of all immediate superior nodes that use this node as a key. *Downlink* points to the linked list of all subordinate nodes. *Key-downlink* points to the linked list of subordinate nodes that serve as keys for the node. *Type* denotes the node type, namely, A, I, G, or M. The field *Level* is the stratified (by algorithm *Labeling-digraph* below) order of the node. The last field *Name* keeps the name of the relation associated with the node.

Listed below are the paths determined by the ESQL processor and their corresponding relational algebraic expressions. For each expression, we will briefly explain the path finding process and the translating process.

1. $A_{d_3}.M_{d_1}$: The path linking object A_{d_3} and property M_{d_1} is $A_{d_3} \rightarrow G_{d_1} \rightarrow A_{d_1} \leftarrow M_{d_1}$. To locate this path, Part II initiates two upward search paths, one from A_{d_3} and the other from M_{d_1}. The two paths meet at A_{d_1} and thus forms a desirable path. Since the path is rather short, $A_{d_3}.M_{d_1}$ can be evaluated by the simple algebraic expression $\pi_{A_{d_3}\#\, M_{d_1}}(r_{A_{d_3}} \bowtie \delta_{A_{d_1}\#\leftarrow A_{d_3}\#} r_{A_{d_1}})$. From left to right, the relation $r_{A_{d_1}}$ first has its attribute $A_{d_1}\#$, the surrogate column, renamed into $A_{d_3}\#$. It then joins (i.e., intersects) with the relation $r_{A_{d_3}}$ and, finally, keeps only the two needed attributes, namely, $A_{d_3}\#$ and M_{d_1}.

2. $A_{d_3}(A_{a_4}, A_{b_1})$: While the previous expression is of the type *object.property* and requires two upward (i.e., backward) paths to form a retrieving path, this expression is of the second type, *event(object,object)*. Since the event node may be a generalization node of several interaction nodes, it must employ also a downward (i.e. forward) chaining to complete the retrieving path. The path thus located, from A_{d_3} down to an individual interaction node, is $A_{d_3} \Rightarrow I_{d_2}$. Knowing that A_{a_5} and A_{b_3} are the two participating objects of I_{d_2}, Part II also initiates two additional search paths; one between A_{a_5} and A_{a_4} and the other one between A_{b_3} and A_{b_1}. The complete path for this expression is therefore $A_{a_4} \rightarrow G_{a_2} \rightarrow A_{a_5} \rightarrow I_{d_2} \leftarrow A_{b_3} \leftarrow G_{b_2} \leftarrow A_{b_1}$. And, the corresponding algebraic expression is $\pi_{A_{d_3}\#\, A_{a_4}\#\, A_{b_1}\#}(\delta_{I_{d_2}\#\, A_{a_7}\#\, A_{b_3}\#\leftarrow A_{d_3}\#\, A_{a_4}\#\, A_{b_1}\#}(r_{I_{d_2}}) \bowtie r_{A_{d_3}} \bowtie r_{A_{a_4}} \bowtie r_{A_{b_1}})$.

3. $A_{a_4}.M_{a_2}$: Similar to expression (1), the path can be located as $A_{a_4} \rightarrow G_{a_1} \rightarrow A_{a_1} \leftarrow A_{a_2} \leftarrow M_{a_2}$. The algebraic expression is $\pi_{A_{a_4}\#\, M_{a_2}}(r_{A_{a_4}} \bowtie \delta_{A_{a_1}\#\leftarrow A_{a_4}\#} r_{A_{a_1}} \bowtie r_{A_{a_2}})$.

4. $A_{c_1}.M_{c_1}$: Again, the path is $A_{c_1} \leftarrow M_{c_1}$. The algebraic expression is $\pi_{A_{c_1}\#\, M_{c_1}}(r_{A_{c_1}})$.

5. $A_{c_1}(A_{a_{10}}, A_{b_1})$: The expression is similar to expression (2) except that the node A_{c_1} is an aggregation node for the generalization node G_{c_1}, which further includes two

subordinate interaction nodes, I_{c_1} and I_{c_2}. The downward path is however only $A_{c_1} \Rightarrow G_{c_1} \Rightarrow I_{c_1}$ because the other one between A_{c_1} and I_{c_2} is identified as an irrelevant path according to the two participating objects. The complete path including the two upward paths is $A_{a_{10}} \rightarrow A_{a_9} \rightarrow A_{a_5} \rightarrow I_{c_1} \leftarrow A_{b_2} \leftarrow G_{b_2} \leftarrow A_{b_1}$. The algebraic expression is $\pi_{A_{c_1}\# A_{a_{10}}\# A_{b_1}\#}(\delta_{I_{c_1}\# A_{a_5}\# A_{b_2}\#\leftarrow A_{c_1}\# A_{a_{10}}\# A_{b_1}\#} r_{I_{c_1}} \bowtie r_{A_{c_1}} \bowtie r_{A_{a_{10}}} \bowtie r_{A_{b_1}})$.

6. $A_{c_1}(A_{a_4}, A_{b_3})$: Similarly, the downward path is $A_{c_1} \Rightarrow G_{c_1} \Rightarrow I_{c_2}$, and the complete path is $A_{a_4} \rightarrow G_{a_2} \rightarrow A_{a_6} \rightarrow I_{c_2} \leftarrow A_{b_3}$. The algebraic expression is $\pi_{A_{c_1}\# A_{a_4}\# A_{b_3}\#}(\delta_{I_{c_2}\# A_{a_6}\# A_{b_3}\#\leftarrow A_{c_1}\# A_{a_4}\# A_{b_3}\#} r_{I_{c_2}} \bowtie r_{A_{c_1}} \bowtie r_{A_{a_4}} \bowtie r_{A_{b_3}})$.

7. $A_{a_{10}}.M_{a_{14}}$: The path is $A_{a_{10}} \rightarrow A_{a_9} \rightarrow A_{a_5} \rightarrow G_{a_2} \rightarrow A_{a_4} \leftarrow A_{a_8} \leftarrow M_{a_{14}}$. The algebraic expression is $\pi_{A_{a_{10}}\# M_{a_{14}}}(r_{A_{a_{10}}} \bowtie \delta_{A_{a_4}\#\leftarrow A_{a_{10}}\#} r_{A_{a_4}} \bowtie r_{A_{a_{11}}})$.

8. $A_{a_{11}}.M_{a_{14}}$: The path is $A_{a_{11}} \rightarrow A_{a_8} \leftarrow M_{a_{14}}$. The algebraic expression is $\pi_{A_{a_{11}}\# M_{a_{14}}}(r_{A_{a_{11}}} \bowtie \delta_{A_{a_{11}}\#\leftarrow A_{a_{11}}\#} r_{A_{a_{11}}})$.

Detailed Algorithm

The previous section presents a number of examples demonstrating some of the typical retrieving paths. It also shows the algebraic expression generated according to each retrieving path. For completeness, this section presents the key algorithms that perform all these tasks. These algorithms, along with the node structure, have been revised several times mainly to reduce the search space of the retrieving paths.

1. Basic functions

Key-subordinates:	the list of nodes indicated by key-downlink of the given node;
Subordinates:	the list of nodes indicated by downlink of the given node;
Key-superiors:	the list of nodes indicated by key-uplink of the given node;
Superiors:	the list of nodes indicated by uplink of the given node.

2. Algorithm *Labeling-digraph*:

procedure:
/ Stratify the hierarchy by assigning each node a level-value */*
Collect all prime nodes (nodes without *superiors*) to form L;
$n := 0$.
  Repeat
    $n := n + 1$;
    for every node $p \in L$, $p.Level := n$;
    Collect all the subordinates of the nodes in L to form a new L;
  Until ($L = \phi$);

end.

3. Algorithm *Path-finding-property*:

input: X, Y (*expression* $X.Y$).
procedure:
/ Find the join path for "Object.Property" and "Event.Property" */*
If $Y \in$ *Subordinates*(X)
then do
begin
/ Trivial cases */*
if *Y.Type* = "M"
then return ("$\pi_{X\#\,Y}(r_X)$")
else return ("$\pi_{X\#\,Y\#}(r_X \bowtie r_Y)$");
end
else do
begin
/ Non-trivial cases */*
L-List := ϕ;
Top-nodes := $\{X\}$;
Repeat
/ Find all its superiors whose properties are inheritable */*
Margin := *MAX* (*p.Level* $\forall p \in$ *Top-nodes*);
Add *Top-nodes* to *L-List*;
Top-nodes := *Key-superiors(Top-nodes)*;
Until (*Top-nodes* = ϕ);
R-List := ϕ;
Top-nodes := $\{Y\}$;
Done := *"false"*;
Repeat
/ Search for the appropriate join with the object/event's superiors */*
For every $p \in$ *Top-nodes*
if *p.Level* $\leq$ *Margin* then delete p from *Top-nodes*;
Join-nodes := *Top-nodes* $\cap$ *L-List*;
For every $p \in$ *Join-nodes*
if *p.Type* = G then delete p from *Join-nodes*;
If *Join-nodes* $\neq \phi$ then *Done* := *"true"*;
Add *Top-nodes* to *R-List*;
Top-nodes := *Superiors(Top-nodes)*;

```
            Until (Top-nodes = φ or Done = "true");
            If Done = "true"
                then do
                    begin
                        /* Identify the join path and return an appropriate algebraic expression */
                        Pick the node with largest Level-value in Join-nodes to be p;
                        expr := r_X ⋈ δ_{p#←X#}(r_p);
                        Join-nodes := R-List ∩ Subordinates(p);
                        Pick the node with largest Level-value in Join-nodes to be p;
                        Do While ( p ≠ Y )
                            begin
                                expr := expr ⋈ r_p;
                                Join-nodes := R-List ∩ Subordinates(p);
                                Pick the node with largest Level-value in Join-nodes to be p;
                            end;
                        if Y.Type = "M"
                            then return ("π_{X# Y}( expr)")
                            else return ("π_{X# Y#}( expr ⋈ r_Y )");
                    end
                else return (nil);
        end;
end.
```

4. **Algorithm** *Path-finding-event*:

```
input: E, O1, and O2 (expression E(O1, O2)).
function Meet(A1, A2):Boolean
    /* Check if the given associations can have common elements */
    begin:
        If A1 = A2 then return ("true");
        If A1.Level > A2.Level
            then do
                begin
                    Margin := A2.Level;
                    Temp-nodes := {A1};
                    High-node := A2;
                end
            else do
                begin
```

```
                    Margin := A1.Level;
                    Temp-nodes := {A2};
                    High-node := A1;
                end;
        Done := "false";
        Repeat
            For every p ∈ Temp-nodes
                if p.Level ≤ Margin then delete p from Temp-nodes;
            If High-nodes ∈ Temp-nodes then Done := "true";
            Temp-nodes := Key-superiors(Temp-nodes);
        Until (Temp-nodes = φ or Done = "true");
        return(Done);
    end;
procedure:
/* Find the join path for "Event(Object1,Object2)" */
I-list := φ;
B-list := {E};
Repeat
    /* Find all the possible "interactions" */
    For every p ∈ B-list if p.Type = "I" then do
        begin
            Delete p from B-list;
            Add p into I-list;
            B-list := Key-subordinates(B-list)
        end;
Until (B-list = φ);
expr := φ;
For every e ∈ I-list do
    begin
        /* Verify the legitimacy of the participants for each "interaction" */
        /* and return an appropriate algebraic expression if it is legitimate */
        P1 := L-participant(e);
        P2 := R-participant(e);
        If Meet(O1, P1) and Meet(O2, P2)
            then expr := expr ∪ π_{E# O1# O2#}(r_E ⋈ (δ_{e# P1# P2#←E# O1# O2#}(r_e)) ⋈ r_{O1} ⋈ r_{O2});
    end;
return(expr)
end.
```

Conclusion and Future Work

With strong expressive power, semantic data models conceptualize more faithfully the underlying application and impose more strict integrity constraints. While it is the complexity of the semantic data model that discourages casual users from using it, it is also the structural information embedded in the semantic model that seems to provide the system with a very useful means to reason. Aiming at providing a more intelligent and user-friendly query language for semantic databases, this research has addressed the problem of automatic path determination. The approach used is a combination of upward and downward chaining of semantic nodes. This work has resulted in an enhanced but simple query language, ESQL, through which query specifications are much more simplified. The syntax of the query language has been designed and tested. The algorithms that find retrieving paths and translate ESQL algebra into relational algebra have been designed. Up to now, all theoretical work has been done. We are currently developing the prototype language.

This work is now being applied to two engineering applications. One is to manage the feature database for a modern mechanical shop. The database is to link all phases of an engineering cycle. The other application is to manage the database for the moat etching process in semiconductor fabrication. The database contains all related information and is mainly to provide more timely corrective fabrication instructions, through queries initiated from sensors as well as operators.

References

[1] D.E. Atkins and R. A. Volz, "Flexible Automatic Discrete Parts Assembly," Annual Report, Center for Research on Integrated Manufacturing, RSD-TR-23-87

[2] R. King and D. J. McLeod, "Semantic data models," in *Principles of database design, Vol. 1, Logical organizations*, ed. by S. B. Yao, Prentice-Hall 1985

[3] Y. C. Lee and M. Doctor, "Automatic Determination of Retrieving Paths in A Semantic Data Model," Proc. IEEE Office Automation Symposium, Gaithersburg, MD, April 27-29, 1987, pp 89-95

[4] Y. C. Lee, C. H. Cho, M. Doctor, and E. Armstrong, "ESQL: An Enhanced but Simple Query Language for A Semantic Data Model," in preparation.

[5] S. Y. W. Su, "Modeling integrated manufacturing data with SAM*," *Computer*, January 1986, pp 34-49

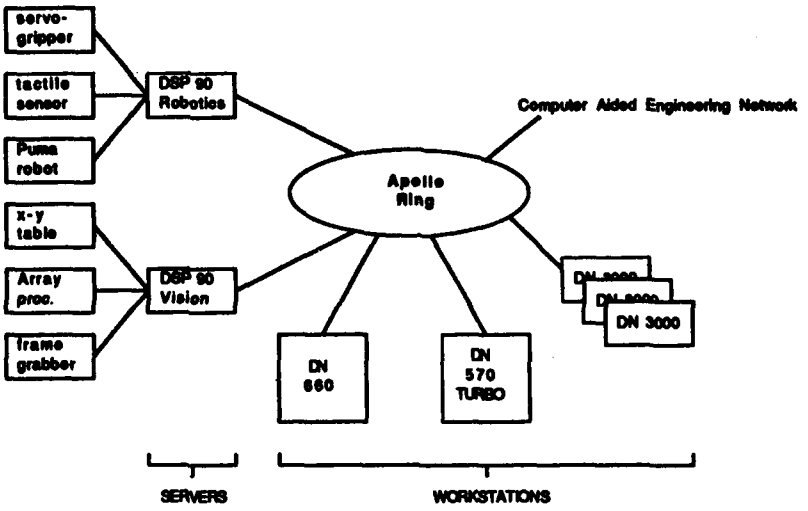


Figure 16: Robot prototyping system.

2.5 System Integration Techniques

2.5.1 *The Robotics Prototyping System*

Investigator: T.N. Mudge

Problem Description and Research Objectives

We have developed a Robotic Prototyping System (RPS) that enables researchers to experiment with ideas quickly. The RPS is an environment in which sensors and effectors are integrated under a common workstation interface. This gives the capability to acquire sensory data quickly from experiments, and the ability to command the effectors from the same programs that acquire and process the sensory data.

Approach

The RPS is based on a network of graphics workstations. Networked graphic workstations support the software development environment on which RPS is based and provide a common interface to the shared resources of the system.

Status

Figure 16 shows the current hardware configuration of the RPS. The system integrates cameras, a Mercury ZIP array processor, a tactile sensor, two PUMA 600 robots with servo grippers, an X–Y table, and a wrist force sensor. During the past year we have developed

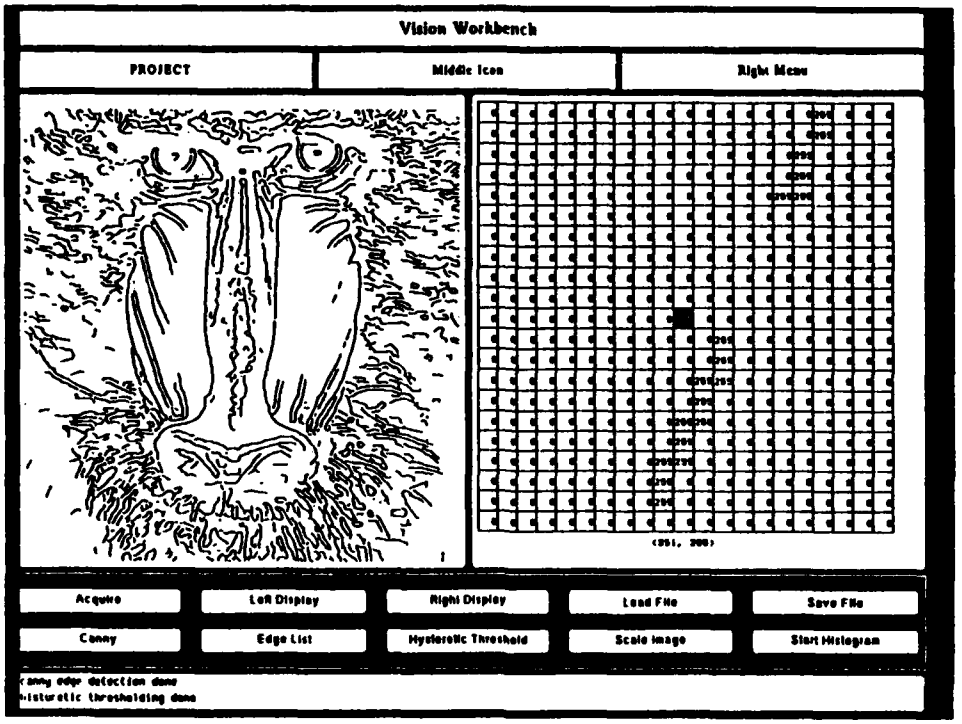


Figure 17: Vision User Interface

reusable software components for the PUMA robots and the Mercury ZIP array processor. These software components provide a base on which researchers can develop higher-level algorithms incorporating these devices. To aid in experimentation a menu and icon user interface built on top of the software components is provided for initial prototyping. With the user interface a researcher can experiment with supplied routines in an interactive fashion using the point and click interface, cut and paste data between applications and save data in files for later use or further processing. The graphics capability of a workstation allows the display of sensory data that has undergone various stages of processing or the 3-dimensional graphical representation of an effector. An example window of the vision user interface is shown in Figure 17. The left panel shows an edge detected image with a 2×2 grid cursor positioned on the lower left side. The right panel shows a blowup of the cursor region that contains the actual pixel values. Such an interface allows a researcher to interactively fine tune the parameters for an edge-detection or any other of the supplied routines, dump these parameters to a file. These parameters can then be used in the supplied routines that are linked with the other parts of the algorithm. This form of rapid prototyping frees a researcher from low level details and facilitates high-level algorithm development.

2.5.2 Research into Vision Algorithms

Investigator: T.N. Mudge

2.5.3 Object Recognition Using Intensity Images

Research Objective

Recognizing objects from intensity images is a task that humans perform routinely. Incorporating such a capability into any manufacturing system would greatly enhance its flexibility and generality. We have investigated object recognition under the assumption of complete knowledge of the geometry of the objects we intend to recognize, i.e., we have geometric models of the objects. Our initial work involved development of a 2-d recognition system possessing a number of novel parts that combine to produce a system that is very efficient and robust. Currently, we are seeking to extend our success with the 2-d recognition system into the 3-d domain. The 3-d problem necessitates the solution to a number of problems that are trivial in the 2-d domain. To this end, we have focused our efforts on the following:

- Finding features for 3-d objects that, in addition to being invariant to image rotation, translation, and scaling, are also robust with respect to image noise and distortion.
- Developing techniques for the robust estimation of the best position and orientation of a hypothesized 3-d model instance in an image.
- Development of *multiview 3-d models* and investigation of optimal methods to determine which viewpoints to include in the models.

Approach

2-d Object Recognition

Our approach to the 3-D problem has been shaped by our earlier, quite successful, work on the simpler problem of 2-D partially visible object recognition. This work is described most recently in [1]. The work that led up to that in [1] includes [2] [3], [4], [5], [6], [7], [8], [9], [10]. In that work, as in the present, we attempted to design a method that wabs as practical, general, and robust as possible. A brief synopsis of it function is given below. A number of its components are used in our 3-D work as well, and as such will be described in greater detail later.

1. An edge detector and edge linker are applied to the image to be interpreted. The edge contours thus obtained are converted to a dual Cartesian and $\theta - s$, or slope angle vs. arclength representation. The Cartesian representation of a contour is simply a list

of the $x - y$ coordinates of each point of the contour in the image plane. The $\theta - s$ representation has arclength measured from a reference point to the point of interest as the abscissa (s) and the angle of the tangent at the point of interest as the ordinate (θ). This representation has a number of useful properties that will be described in more detail later.

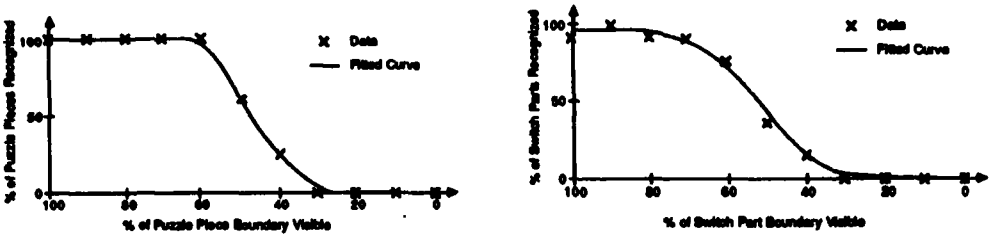
2. Critical points are detected, and CPN features are extracted. A critical point is a point of extremal curvature in the contour. Since the derivative of the $\theta - s$ representation with respect to arclength gives curvature, detecting critical points amounts to applying a one-dimensional edge detector to the contour. A *critical point neighborhood* (CPN) feature is the result of windowing out a section of the $\theta - s$ contour such that a critical point is centered within it, and then shifting this section of the curve so that the value of θ at the critical point is zero. This normalizes the CPN feature so that it is invariant to rotation and translation.

3. The CPN features, which may be viewed as vectors of some dimension, N, are projected onto the subspace that contains most of the variance of the CPN features that are expected to appear in the images to be presented to the system. The basis of this subspace is determined offline using the Karhunen-Loève expansion on the model contours. In our experiments, the feature vectors were reduced from forty five elements to five elements.

4. The five element feature vectors are used to query an advanced data structure, called a k-d tree, which allows us to retrieve all model feature vectors (which have been precomputed from the model contours, and projected onto the Karhunen-Loève subspace) that are nearby in the 5-D space. One important point is that this retrieval can be done in $O(\log N)$ time, where N is the number of features. This fact, coupled with the data reduction and certain properties of the projections, give the algorithm its speed.

5. Each precomputed model feature that is retrieved from the k-d tree is a hypothesis, or conjecture, as to the identity of the object that gave rise to the image feature. This is so because each model feature that is returned is associated with one and only one model. In addition, each match specifies a transformation from the model coordinate system to the viewer's coordinate system, namely the transformation that bring the model and image feature into the closest correspondence. It now remains to verify whether the conjecture is right or wrong. This is done by making a decision based on the following two metrics for measuring the goodness of a match:

    - S_{cp} – the fraction of critical points predicted by instantiating the hypothesized model using the transformation predicted by the matching features.



(a) (b)

Figure 18: In (a) a plot of the results of running our algorithm on a set of images containing puzzle pieces is shown, while in (b) are the results of running the algorithm on a set of images containing microswitch parts. Each plot is a graph of the percent of objects recognized correctly versus the fraction of their boundary that is visible.

- S_b – the fraction of boundary predicted as in the above case which actually is found to appear in the image.

If both S_{cp} and S_b are found to be near enough to unity, then the conjecture is accepted as true. If not, then it is rejected as false. The conjectures accepted as true are the system's best interpretation of the image.

Figure 18 shows the results of running the 2-D algorithm on images of two sets of parts. Ten jigsaw puzzle pieces comprised the first set of parts, while a microswitch assembly comprised the second set. Each image contained at least nine objects, randomly placed in a pile. The tests were run on an Apollo DN570, a 5 VAX MIP machine. The average time for the algorithm to completely interpret the image (i.e., at least nine parts) on the DN570 was about 1.5 seconds. Based on tests with the jigsaw puzzle pieces, people can perform the same task in about 20 seconds after being thoroughly familiarized with the parts.

3-d Object Recognition

Features for 3-d Recognition: Scale-Invariant Critical Point Neighborhoods

As in the 2-D algorithm, we detect critical points in from the $\theta - s$ contour. In contrast to the 2-D method, we use a sophisticated multi-resolution edge detector which is based on the method described in [11]. This greatly improves the localization of the critical points over that obtained from a single resolution resolution edge detector. A line is then fit to the largest number of points near the critical point such that the residue of the fit does not exceed a limit χ_L. This insures that we do not attempt to fit the $\theta - s$ curve where it is not roughly linear. Let θ_{cp} and s_{cp} specify the location of a critical point on a $\theta - s$ contour. Using the equation of the fitted line, it is now a simple matter to find the points where the line exceeds $\theta_{cp} + T$ and falls below $\theta_{cp} - T$, for some offset T. Call the values of arclength

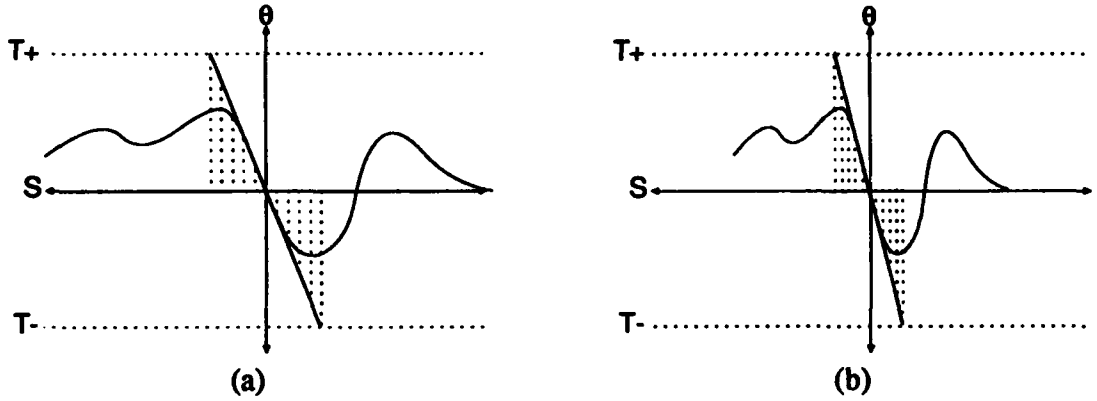


Figure 19: This figure illustrates the process of computing a SICPN. In (a) is shown an unscaled $\theta - s$ contour, and in (b) the same contour is scaled. In each case, the values of θ where the dotted vertical lines intersect the $\theta - s$ curve form the components of the SICPN vector.

at these points s_- and s_+. Resampling the $\theta - s$ curve between s_- and s_+ leads to a feature vector that is invariant to scale. Scale invariance follows from the fact that when a contour is scaled, its $\theta - s$ representation scales along its abscissa, s. Since the slope of the fitted line is inversely proportional to scale, it is easy to show that the spacing between the sample points is proportional to scale, implying that θ at the sample points is constant. The quantity T defines that amount of contour that is included in the SICPN feature. Figure 19 illustrates the process of deriving a SICPN from a $\theta - s$ contour. Figure 20 shows some SICPNS from two objects that differ roughly by a factor of 2 in scale.

Estimation of Position and Orientation: Feature Modulated Attractors

One of the key components of any 3-d object recognition algorithm is the matching module. The matching module, in turn, usually consists of two major components: a type of *associative memory* that quickly generates plausible model instantiations based on the features observed in the image, and a *hypothesis extender* that reexamines, employing additional evidence from the image, the likelihood that a hypothesized model instance indeed is present in the scene. The 2-d method employed a k-d tree as a means of implementing an associative memory which retrieves models possessing similar local shape [1]. In this paper, we address the second part of the problem by a novel approach that we call *feature modulated attractors*. In order to do detailed evaluation of the merit of a hypothesized instance, its attitude must be determined so that it best explains the image data. Previous methods to compute the best attitude of a model have relied on the simplicity of the features used (e.g., line segments or groups of line segments), or have relied on unrealistic assumptions about the image formation process (e.g. ignoring the effects of self occlusion). The feature modulated attractors (FMA) approach makes only the assumption that the numerical attributes of features (such as the

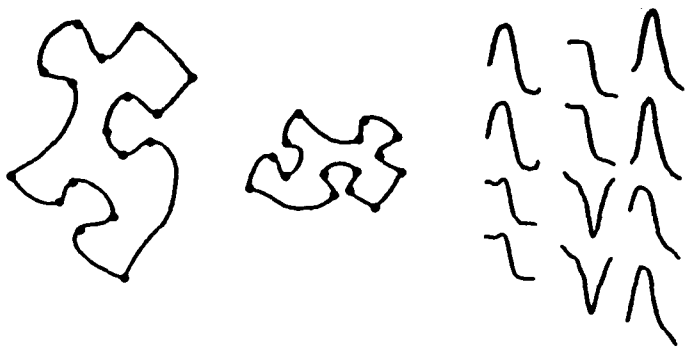


Figure 20: Each pair of SICPN vectors is comprised of elements from corresponding critical points from two images of objects. Of each pair, the SICPN's on the top is from the image of the larger version of the objects, while that on the bottom is from the image of the smaller.

angle between two portions of an edge on an object) is a piecewise smooth function of the viewing parameters, over the domain that the feature is visible. The FMA method is based on the intuitive notion that 1) predicted model features that have similar descriptive attributes (we use shape attributes, although any kind of feature attribute could be used); 2) should be "attracted" by like features in the image; and 3) that the viewing parameters of the hypothesized model instance should be adjusted to respond to these forces. We achieve this goal by minimizing an energy potential function resulting from the "forces" exerted on the hypothesized features (computed from the model) by the observed features (computed from the image). The FMA method also has the advantage that no feature correspondences are hypothesized, eliminating the need for a backtracking search to rectify incorrect hypotheses. In addition, since the forces generated by the attractors decay with distance (the rate is controllable by higher level processes) the FMA method is fast since a small fraction of the features in the image are involved in generating the force acting on a hypothesized feature. Figure 21 illustrates the concepts involved with the FMA approach. There, a 3-d model of a tilted jigsaw puzzle piece with its object centered coordinate system is shown at (a). The 3-d model is being projected onto a viewing plane, (b), according to the current viewing parameters. The actual image plane, (c), contains both an instance of a puzzle piece and a scissors-like object. The image plane and the model projection plane share the same coordinate system, and are actually identical; we have separated them for clarity. The dots on the image of the puzzle piece, (e), represent corner-like critical point neighborhood [1] features. The property attributes of the features encode the local shape of the edge contours. The feature marked by a diamond in (d) is being attracted to varying degrees to all of the

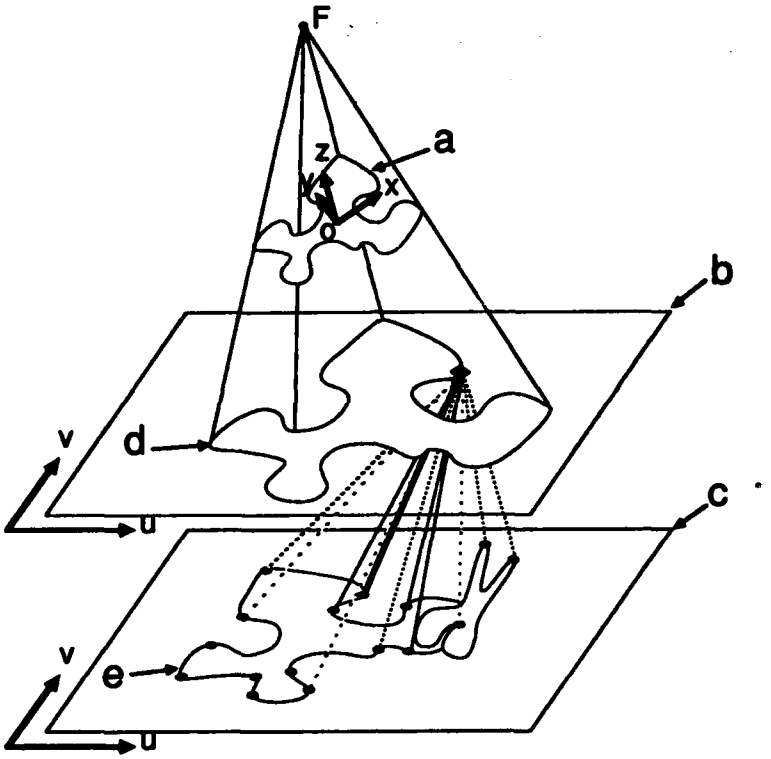


Figure 21: An illustration of a SMA approach.

observed features in the image, as indicated by the lines between the feature in (b) and those in (c). The density of the lines indicate the strength of the force. Many of the forces are weak because of large differences in local shape, or in position and orientation. The thick line is a very strong force acting between the model-generated feature and its correct counterpart among the observed features. The reason this attraction is so strong is due primarily to the similarity in shape and to a lesser extent to their relative proximity. If all the features on the model are being similarly attracted, and the model is allowed to fall into equilibrium, then many of the model-generated features will be in close alignment with their correct counterparts among the observed features.

Status

The algorithm resulting from our research on the 2-d problem was successful enough that we consider this problem completely solved. The 3-d portion of the project represents ongoing research. The scale-invariant CPN features described above have been characterized; the results of this work will be given at the SPIE Cambridge Symposium "Advances in Intelligent Robotics Systems. The Feature Modulated Attractor technique is being implemented and is nearly ready for experiments on real data to commence. Investigation of the FMA

method will be funded in the near future by CAMRSS. Investigation of issues involved in the modeling of objects is also under continuing investigation.

References

[1] Gottschalk, P. G., J. L. Turney, and T. N. Mudge. "Two-Dimensional Partially Visible Object Recognition Using Efficient Multidimensional Range Queries," *1987 IEEE Conference on Robotics and Automation*, pp. 1582-1589.

[2] Mudge, T. N., J. L. Turney, and R. A. Volz. "Automatic Generation of Salient Features for the Recognition of Partially Occluded Parts," *Robotica*, Vol. 5, 1987, pp. 117-127.

[3] Turney, J. L., T. N. Mudge and R. A. Volz. "Solving the Bin of Parts Problem," *Proceedings of Vision '86*, (a Machine Vision Association of the SME Conference and Exposition), Detroit, MI, June 1986, pp. 4-21 through 4-38.

[4] Dolezal, R. M., T. N. Mudge, J. L. Turney and R. A. Volz. "Determining the Pose of an Object," *Proceedings of the Society of Photo-optical Instrumentation Engineers 2-nd International Symposium on Computer Vision for Robotics*, SPIE Vol. 595, Cannes, France, December 1985, pp. 68-71.

[5] Turney, J. L., T. N. Mudge and R. A. Volz. "Recognizing Partially Occluded Parts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-7, No. 4, July 1985, pp. 410-421.

[6] Turney, J. L., T. N. Mudge and R. A. Volz. "Recognizing Partially Hidden Objects," *Proceedings of the IEEE International Conference on Robotics and Automation*, March 1985, pp. 48-54.

[7] Turney, J. L., T. N. Mudge and R. A. Volz. "Recognizing Partially Hidden Objects," *Proceedings of the Society of Photo-optical Instrumentation Engineers Cambridge Symposium on Intelligent Robots and Computer Vision*, November 1984, pp. 108-113.

[8] Mudge, T. N., J. L. Turney and R. A. Volz. "Experiments in Occluded Parts Recognition," *Proceedings of the Society of Photo-optical Instrumentation Engineers Cambridge Symposium on Intelligent Robots*, SPIE Vol. 449 (Part 2), November 1983, pp. 719-725.

[9] Mudge, T. N., and T. S. Abdel-Rahman. "Case Study of a Program for the Recognition of Occluded Parts," *Proceedings of the 2nd Annual IEEE Computer Society Workshop on Computer Architecture for Pattern Analysis and Image Data Base Management*, Pasadena, CA, October 1983, pp. 56-60.

[10] Turney, J. L., T. N. Mudge, R. A. Volz and M. D. Diamond. "Experiments in Occluded Parts Recognition Using the Generalized Hough Transform," *Proceedings of the Conference on Artificial Intelligence*, Oakland University, Rochester, MI, April 1983.

[11] Schunck, B. G. "Gaussian Filters and Edge Detection," *General Motors Research Publication No. GMR-5586*, pp. 33-47, October 20, 1986.

2.5.4 Heuristic Search

Investigator: T.N. Mudge

Problem Description and Research Objectives.

Heuristic search is a fundamental solution technique to many problems of optimization in robotics, vision and artificial intelligence. These problems are characterized by not having direct solution methods. Solutions are found by searching through the solution space of a problem until a desired solution is found. Heuristic searchs require an enormous amount of computation. This is particularly true since many of the problems are NP-hard and the computational effort to conduct the search grows exponentially with the size of the problem.

An attractive solution to deal with the enormous computation requirements of search problems is parallel processing. Although parallel processing cannot overcome the exponential growth associated with most search problems, considerable speedups can be obtained. Furthermore, the the emergence of very high performance 32-bit microcomputers with the functionality and performance of supermini class computers makes parallel processing more attractive. Assembling large numbers of these to form massively parallel machines is now a possibility. The objective of this research is to investigate the uses of massively parallel processors for search problems in particular those that relate to vision applications. Figure 22 shows an example of using backtracking in template matching [1].

Approach. Our work in this area has focussed on developing algorithms for the NCUBE massively parallel hypercube multiprocessor [4]. An n-dimensional hypercube or n-cube computer is a multiprocessor that consists of $N = 2^n$ processors interconnected as an n-dimensional binary cube. Each processor P forms a node of the cube with its own cpu and local main memory. P has direct communication paths to n other processors (its neighbors), which correspond to the edges of the cube that are connected directly to P. Figure 23 illustrates the hypercube topology for $n = 4$, or a 4-cube. Hypercubes have a number of features which make them attractive for vision applications as well as many other applications [2], [3], [4].

The first step in this work is to characterize search algorithms. Our initial attempts are examining generalized branch and bound (B&B) algorithms. A large class of algorithms

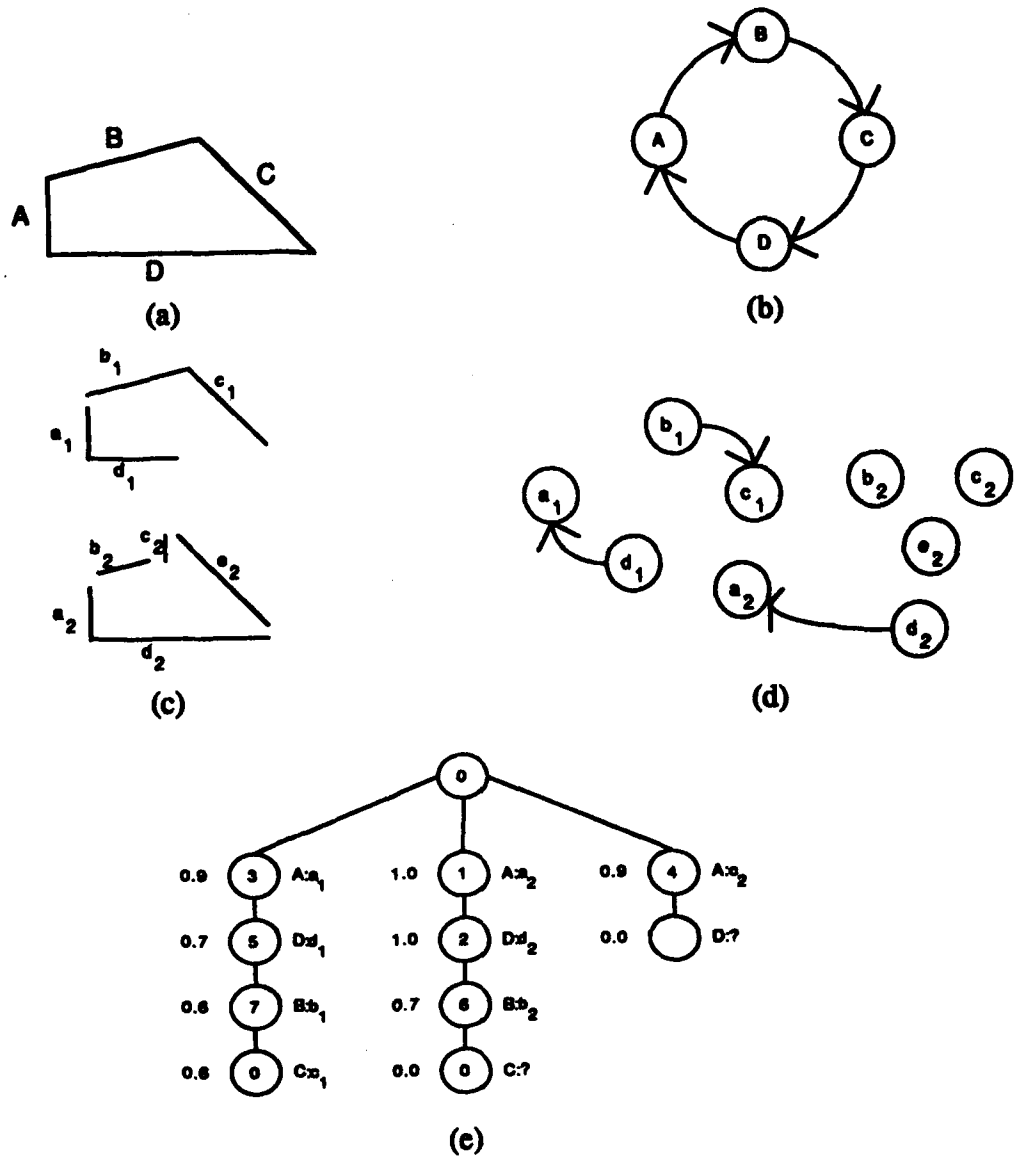


Figure 22: An example of search in vision.

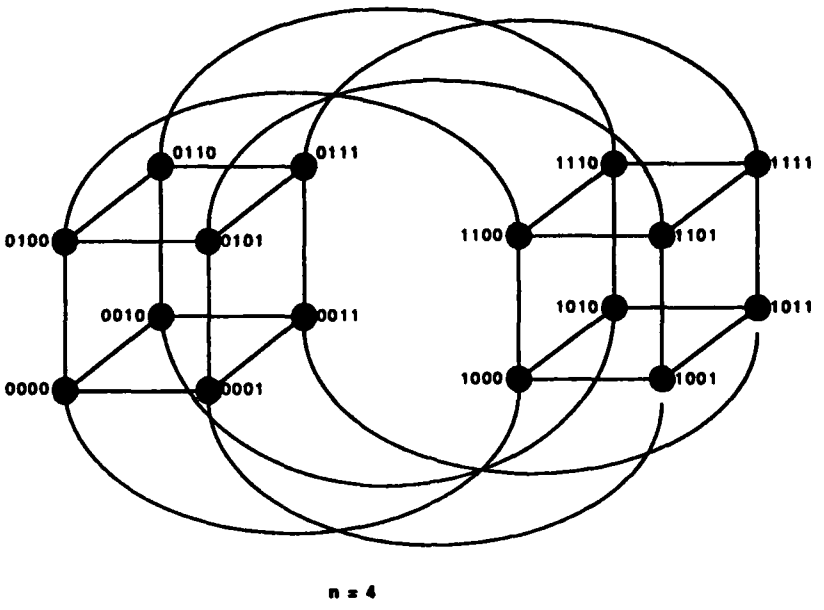


Figure 23: A 4-cube.

such as graph theoretic algorithms, backtracking, searching, AND-OR search, and rewriting systems can be viewed as B&B, and these are the kinds of algorithms that one associates with "reasoning" about a scene, particularly at the level where multisensor integration is being performed.

The B&B algorithms finds a solution to a problem by repeatedly decomposing the solution space of that problem into subspaces of decreasing sizes examining each subspace until a solution is found. To bound the number of subspaces examined, if a subspace is shown not to contain a solution it is eliminated from further decompositions by the algorithm. The repeated decomposition and elimination of solution subspaces forms the "branching" and "bounding" of the B&B algorithm. The decomposition process is represented as a rooted tree over the solution space. The root of the tree represents the complete solution space. Children nodes represent partial solution spaces of the original space, or equivalently *subproblems* of the original problem. The algorithm selects a subproblem that is most likely to lead to a solution based on some heuristic information from the problem domain and decomposes it to smaller subproblems, hence, extending the B&B tree.

A B&B algorithm requires global information for the selection and deletion of subproblems during the search process. In the case of distributed memory multiprocessors such as the hypercube, enforcing this requirement can result in severe performance degradation due to the high communication overhead among the processors. An interesting property of

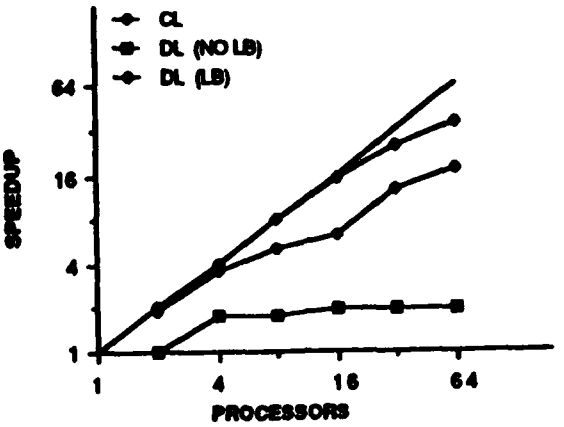


Figure 24: Results.

parallel B&B algorithms is that such global information may be only partially maintained at the expense of more search steps to find a solution. In other words, the lack of the global information will not affect the correctness of the algorithm but will only affect its efficiency.

Status. We have considered this possible tradeoff between the communication overhead and the number of solution steps in our experiments. Figure 24 shows the speedup of two implementations which illustrate the tradeoff [5]. The first algorithm, referred to a the centralized algorithm (CL), maintains all global information in the host memory. In this case, all the information is available to the processors, but rather at a high communication overhead. The second algorithm referred to as the distributed algorithm (DL), distributes the global information across the processors. Consequently, not all of the information is accessible to all the processors in the system. However, the communication overhead is low. A load balancing scheme is employed to balance the subproblems across the processors of the hypercube and attempt to reduce the amount of additional search steps which result as a consequence of the distribution of the information [5]. The results indicate that distributing the information can lead to better results only with the load balancing. The results also indicate that without the load balancing the distributing the information can lead to severe degradation in performance. This degradation can be attributed to two factors. First, processors can become idle and do not participate in the search process. Second, processors can search unnecessary subspaces due to the lack of global information. Our load balancing attempts to deal with these two factors and improves the performance. The performance of these algorithms can be compared to the performance of other systems as shown in Figure 25. The NCUBE figures are for 64 processors and show a speedup over both the VAX 11/780 and the IBM 3090.

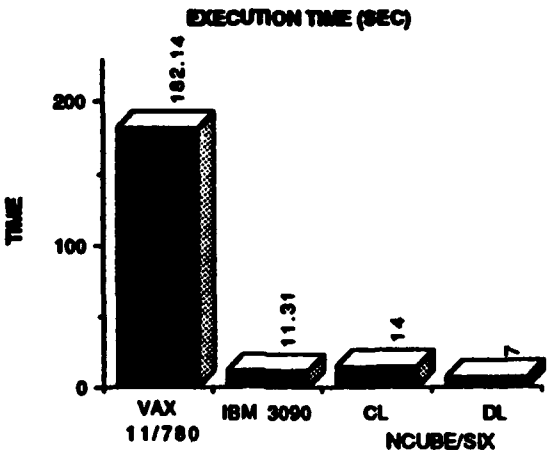


Figure 25: Performance.

References

[1] T. N. Mudge and T. S. Abdel-Rahman. "Architectures for Robot Vision," in *Specialized Computer Architectures for Robotics and Automation,* J. Graham, ed., Gordan & Breach, New York, 1987.

[2] T.N. Mudge and T. S. Abdel-Rahman. "Vision Algorithms for Hypercube Machines," *Journal of Parallel and Distributed Computing*, 4, 1987.

[3] T.N. Mudge. "The Next Generation of Hypercube Computers," *ARO Workshop on Future Directions in Computer Architecture and Software,* May 1986.

[4] J.P. Hayes, T.N. Mudge, Q.F. Stout, S. Colley and J. Palmer. "Architecture of a Hypercube Supercomputer," *1986 International Conference on Parallel Processing,* August 1986.

[5] T. S. Abdelrahman and T. N. Mudge. "Parallel Branch and Bound Algorithms on Hypercube Multiprocessors," *Proc. of the Third Conference on Hypercube Concurrent Computers and Applications,* G. Fox, ed., ACM, New York, 1988.

2.5.5 Distributed Systems Integration Techniques

Investigator: R.A. Volz

Due to the importance of related work on distributed systems integration techniques to this project, we briefly describe that work. Essentially, this work revolves around distributed computing techniques.

Problem Discussion

Distributed computing is an essential part of most large systems today. Until now, however, the principal focus on distributed computing systems has been on their architecture (particularly interconnection mechanisms) and their gross capabilities (usually calculated in some simplistic sense such as multiplying the capabilities of a single processor by the number of processors in the system). One of the most critical problems for the future is the set of programming tools available for such systems.

Software tools for distributed systems must deal with both diverse hardware and the use of existing software written in a wide variety of languages. They must also incorporate the best techniques developed in software engineering over the past two decades and extend these concepts effectively for use in distributed systems[9] It is our hypothesis that extending these concepts to permit distributed execution is a critical step in addressing the distributed computing software problem. Two obvious benefits of so doing are the extension of compile time error checking across machine boundaries and allowing the programmer to use normal language mechanisms for expressing parallel (concurrent) operation without having to invent new application level communication protocols. (Note that computer communication networks do *not* adequately address the applications level protocol issue.) These two advantages alone should greatly improve the problem of developing software for distributed systems.

Summary of Previous Status

The implementation of a system to support distributed program execution for real-time applications requires the solution to a substantial number of problems. Previously, we have discussed the following basic issues:

- Problem analysis—the basic issues in distributed program execution, [7],
- Timing mechanisms—basic software timing mechanisms amongst distributed tasks, [1], and new instruction level mechanisms for simplifying timing implementation, [8],
- Performance evaluation—specifically oriented toward the real-time performance of emitted code, [2],
- Experimental implementation techniques, [3, 4], and

[9]Among the key concepts are: (1) data encapsulation and hiding, (2) abstract data types, (3) modularization of programs, (4) separate compilation (of both modules and specifications), (5) concurrency mechanisms at the language level, and (6) extensive compile time error checking.

- Application of these ideas to manufacturing software, [5].

The capabilities of the translation system at the time of the last report included the distribution of library packages and subprograms with remote access available to:

- both simple and complex data objects. Record and array objects can be nested arbitrarily deeply within one another and include pointers to remote objects.
- subprograms
- declared task objects (no timed or conditional calls across processor boundaries).

The distribution is accomplished statically among a set of homogeneous processors. There is also a Unix make-file like capability to simplify use of the translation system. Tests have been successfully completed with up to three VAX processors cooperating on the execution of a single program.

Progress During the Past Year

During this past year, we have concentrated on three things, the continued development of techniques for our second generation translator, porting it to a network of IBM PC/AT computers, and further application of these ideas to manufacturing software, the latter work heavily involving Prof. A. W. Naylor and being partially funded by General Motors.

Translator Status and Issues

During the past year, we have dealt with the following issues in the translation system:

- Distribution of task objects created from task types,
- Instantiation of generics on distributed processors,
- Renaming,
- Modifying the underlying mailbox system to use interrupts, and
- revising the type handling mechanism to be more efficient and accept the more general initialization and default specifications.

Presently, the system allows task objects created from task types (in the specification of a library package) to be placed on any site in the system. The handling of generics is nearly completed. And, renaming works for all of the more common situations. Those remaining are logically straightforward, but require substantial effort to implement.

Generics and task types raised basically the same problem: One does not know at the time they are submitted for compilation which object references are local and which are remote. While it would be possible to utilize a general addressing mechanism that checks upon each reference whether it is local or remote, the associated overhead would be prohibitive for those that are, indeed, local. The problem is most easily seen with an example. Consider the following (recall that we use a **pragma** called SITE to specify the location on which each library unit is to execute.):

```
pragma SITE (2);
package A is
    task type T is
        entry E(..);
    end;
      ⋮
end A;

with B;
package body A is
    I: INTEGER;
    task body T is
          ⋮
    begin
          ⋮
        B.P;
          ⋮
        I := .. ;
          ⋮
    end T;
          ⋮
end A;
    ⋮
pragma SITE(1);
with A;
procedure M is
T1: A.T;
  ⋮
begin
```

```
   :
end;
```

The task object T1 created on site 1 (in procedure M) references procedure P in package B. Suppose, now, that procedure M is submitted for compilation after package body B. Then the site of the procedure using task type T is unknown. Hence, it is unknown whether package B is on the same site as the procedure that will use T or not. Thus, it cannot be known at compile time whether the reference B.P is local or remote. Worse, since the task type T may be used by several different types, the reference may be local on one and remote on others. Further, package B may include yet other packages or subprograms. Of course, there can be many such packages referenced by the body of A.

Obviously, then, multiple versions of the task body are required, unless one is willing to accept the overhead of generalized addressing. Still, one does not necessarily know at the time that the body of A is submitted for compilation how to compile task body T. The answer is deferred compilation of copies of the task body T. Each time a unit is compiled that creates an object of type T, a check is made to see if an instance of T has yet been created for the site upon which the unit will reside; if not, a new version of T is created for that site. Fortunately, the complexity of the number of copies of task body T that must be generated is linear in the number of sites in the system.

The above example exposes another issue in the implementation of distributed task objects from task type. The hidden variable I, in the body of package A, becomes a shared variable across the network. Moreover, the remote accesses to it from the various task objects created on the network are hidden. That is, the programmer does not necessarily know that they are there. This latter fact may or may not be important, but if the programmer is developing a real-time system, the timing difference between local and remote accesses may well be important. There is no resolution to these issues within the current language definition.

There is no particular difficulty in handling renaming clauses. However, due to the large number of ways in which renaming can be utilized, the modifications to the translator required to handle all of them are extensive and have little to do with distributed execution. Since our primary intent is learn more about distributed execution, we have chosen to handle the more common cases and ignore (at least for the time being) the rest.

The initial versions of the underlying Ada compiler available to us did not support interrupts. Thus, our initial version of the mailbox used a polling scheme for communication. Shortly before the beginning of the project year we received an upgraded compiler that does support interrupts. A revised system using the interrupts was created. This reduced our communication times somewhat, though not as much as we had anticipated. Benchmark tests were developed to study the communication times. It was found that the Ethernet message times were much longer than expected, being on the order of 30 ms, roundtrip.

This completely swamps any overhead introduced by our translation system, and remote rendezvous times are thus also on the order of 30 ms.

A more significant problem with interrupts arose due to the implementation of the interrupts. The Ada reference manual allows two forms of interrupt handling, those that queue pending interrupts (and correspond to normal task entry calls), and those that do not queue the interrupts (corresponding to conditional entry calls). Unfortunately, the underlying compiler implementation with which we were working used the latter implementation, and from time to time, interrupts, and hence messages, are lost. This problem is serious, but not one of the translation system. Rather, it is one of the underlying network and compiler used. We thus, do not consider it further here.

A final problem that arose during the year relates to initialization of variables or default variables in records. An example will clarify the situation.

```
pragma SITE(1);
with A;
with B;
procedure M is
    X: A.R;
    ..
begin
..
    B(..);
end;
```

```
pragma SITE(2);
with A;
procedure B is
    Y: A.R;
    ..
begin
    ..
end;
```

```
pragma SITE(3);
with C;
package A is
    type R is
        record
        U: INTEGER := C.F(..);
        ..
    end record;
    ..
end A;
```

Suppose that the function C.F(..) has side effects. Then the order in which the calls are made can affect the value returned. On a single processor the elaboration order would be C, A, B, M. However, in the distributed system, this elaboration order need not necessarily occur unless extra controls are developed and the wrong values could be placed in the variables. This suggests strongly that initializing things using functions having side effects is very poor and should be eliminated. Moreover, in our particular implementation, we replicate the types part of distributed packages on the sites that use it. That exacerbates the problem considerably. We have found that it is possible to get around this difficulty, but it involves the creation of a number of additional support packages and extra variables we call *shadow* variables. The operational overhead is not high, but the complexity of the translation system is increased substantially. A better solution, we believe, is to modify the language to prohibit initialization of variables or defaults using functions with side effects. Henceforth, we shall assume such a limitation in our implementation.

Through the generosity of the Verdix Corporation, the distributed translation system was demonstrated at the Ada Expo held in Boston, Mass. in December of 1987. Two different

demonstrations were given. The first involved calculating and displaying Mantelbrot sets. An Ada program was written using concurrent tasks to perform these calculations and display a map of the complex values computed. This was displayed in a window on a single Sun computer. At the same time, the program was distributed across two Sun computers, using our SITE **pragma**, and displayed in another window. The difference in speed was very evident, as expected. In the second demonstration, a mobile vehicle with a television camera mounted on it was set up to follow a track around the end of the booth. The vision processing from the camera was done on one Sun computer, while the vehicle processing was done on another.

Port to PC

A second major activity during the year was to port the Distributed Ada system to a network of IBM PC/AT's. In this case the Alsys compiler was used. Although the port was successfully completed, several major difficulties arose, and there are limitations (temporary we believe) that make the system less useful than at first expected. We had originally planned to utilize the network of distributed Ada PC's for implementation of a tele-autonomous system. At present the tele-autonomous system, while implemented in Ada, is done with separate programs running on the different machines.

The first difficulty that arose, and one that had nothing to do with the translation system was the interface to the network. The software to drive the Ethernet boards is written in C. Since both Ada and C do resource management, conflicts arise and getting the two work together is much more than just a matter of matching calling sequences. Moreover, although the Alsys compiler provides an INTEFACE **pragma** for C, it is necessarily implementation dependent, and unless the particular C interfaced to is the same one as used for the Ethernet board drivers, one still does not have a match. It took six months to resolve this problem.

The choice of C as the implementation language for Ethernet board drivers has been made by nearly all of the Ethernet vendors. This decision has consequences that probably were not envisioned (and would not be present if assembly language had been used). It makes it *very* difficult to have any language other than C use the Ethernet in anything other than superficial ways. Moreover, since the C libraries often make use of the underlying operating system, it can make the Ethernet boards useless for embedded systems.

The limitations to our implementation are twofold. First, and more serious, we use the interface to the Ethernet in sophisticated ways, and have uncovered bugs in both the Ethernet board drivers and the Ada compiler's interface to C. Until these are fixed by the vendors (not yet done) we can only operate the network using the polled version of our mailbox system. Second, due to a limitation of Ada programs to the first 640 K of memory in the PC/AT, our translation system can only handle very small programs (since it, itself, takes up most of the memory and leave little working storage). This second problem is not severe, however, because it is easy to cross translate from the Sun.

Manufacturing software

Our work on both the theoretical and the practical aspects of manufacturing software has continued. A distributed factory system has been implemented to illustrate our basic methodology for designing and implementing control software for manufacturing systems, [5].

Our methodology starts by creating a prototype of the manufacturing system. This prototype captures the formal models of components on the factory floor. These models are based on extensions to first-order logic and are composed of a set of first-order predicates that capture the system's configuration while in operation. The values of these predicates are manipulated by a set of rules that model the behavior of the system's components. A rule is enabled when its precondition is satisfied by the present configuration of a system. Enabled rules that pass the conflict resolution filtering process will fire in parallel resulting in a new system configuration. The prototype of a moderate size manufacturing system might result in a large set of rules. The theoretical work was concentrated on detecting any inconsistencies in writing these rules. In particular, Prof. Naylor and his students have derived solutions for the updating, continuity/consistency and existential quantifier problems that are encountered when parallelism is allowed in rule firing.

Once a final prototype has been validated, our methodology develops an Ada implementation of the system. The manufacturing system components are implemented by a set of reusable Ada packages. Reusability is achieved by decoupling the services offered by a given component from the control strategy used in integrating these services with the services of other components to perform a given manufacturing task. The Ada implementation is next distributed using the Ada translator.

Future Directions

Both areas of work described above will continue after the conclusion of this project. A new project with NASA in the Distributed Ada area will extend this work. We expect to complete the implementation of generics in the near future and to begin studying the implications of Distributed Ada implementations for tightly coupled architectures mixed with loosely coupled architectures. As noted in our paper [7], the system architecture is one of the basic dimensions affecting a distributed execution system. Thus, a number of different issues must be considered in the translator design.

In the manufacturing software area, a project with General Motors will apply the ideas to an actual test cell being developed within GM. A simulation of the system will be done first. Then, selected simulated components will be replace with the real components. This is actually a joint project, with personnel from both groups working together on it.

References

[1] R. A. Volz, T. N. Mudge, "Instruction Level Timing Mechanisms for Accurate Real-Time Task Scheduling," *Proc. of the IEEE 1986 Real-Time Symp. and Special Issue IEEE Trans. on Computer*, Vol. C-36, No. 8, August 1987.

[2] R.M. Clapp, L. Duchesneau, R.A. Volz, T.N. Mudge, T. Schultz, "Toward real-time performance benchmarks for Ada," *Communication of the ACM*, Vol. 29, No. 8, pp. 760-778, August 1986.

[3] R.A. Volz, T.N. Mudge, A.W. Naylor, J.H. Mayer, "Some problems in distributing real-time Ada programs across machines," *Ada in use, Proc. of the 1985 Int'l Ada Conf.*, pp. 72-84, May 1985.

[4] R. Volz, P. Krishnan, R. Theriault, "An approach to distributed execution of Ada programs," *1987 IEEE International Symp. on Intelligent Control*, Philadelphia, January 20, 1987.

[5] A. W. Naylor, R. A. Volz, "Design of Integrated Manufacturing System Control Software," *IEEE Trans. Systems Manufacturing and Cybernetics*, Vol. SMC-17, No. 11, Nov/Dec 1987.

[6] A.W. Naylor, R.A. Volz, "Software for integrated manufacturing systems, Part I and II," *Space Operations Automation and Robotics 1987 Conference*, Houston, TX, August 1987.

[7] R.A. Volz, T.N. Mudge, G.D. Buzzard, P. Krishnan, "Translation and Execution of Distributed Ada Programs: Is It Still Ada?," *Special Issue on Ada of the IEEE Transactions on Software Engineering*, 1987.

[8] R.A. Volz, T.N. Mudge, "Timing Issues in the Distributed Execution of Ada Programs," *IEEE Transactions on Computers*, Vol. C-36, No. 4, April 1987.

3 Publications

3.1 Journal Publications

1. "Report of the Robot Programming Language Working Group: NATO Workshop on Robot Programming Languages", R.A. Volz, *IEEE Journal of Robotics and Automation*, Vol. 4, No. 1, February 1988

2. "Time-Optimal Control for a Robotic Contour Following Problem," H.-P. Huang, N.H. McClamroch, *IEEE Journal of Robotics & Automation*, Vol. 4, No. 2, April 1988.

3. "Design of Integrated Manufacturing System Control Software," A.W. Naylor, R.A. Volz, *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. SMC-17, No. 11, Nov/Dec 1987.

4. "Using ADA as a Programming Language for Robot-Based Manufacturing Cells, R.A. Volz, T.N.Mudge and D.A. Gal, *Control and Programming in Advanced Manufacturing*, edited by K. Rathmill, IFS Ltd, UK, Springer-Verlag

5. "A Variational Dynamic Programming Approach to Robot-Path Planning with a Distance-Safety Criterion", S.H.Suh and K.G. Shin, *IEEE Journal of Robotics and Automation*, Vol. 4, No. 3, pp. 334-349, June 1988.

6. "Motion Stereo Using Ego-Motion Complex Logarithmic Mapping," R. Jain, S.L. Bartlett, N. O'Brien, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-9, No. 3, pp. 356-369, May 1987.

7. "Some Experiments in Ego-Motion Comples Logarithmic Mapping", R. Jain, S. Bartlett, and N. O'Brien, *Advances in Computer Vision and Image Processing*, Vol. 3, (ed. T.S. Huang), pp. 145-177, JAI Press, 1988.

8. "Port Manipulator for the Distributed Realization of an Integrated Manufacturing System," K.G. Shin and H.K. Lee, *Computer Systems Science and Engineering*, Vol. 3, No. 1, pp. 21-31, January 1988.

9. "Robust Trajectory Planning for Robotic Manipulators under Payload Uncertainties," K.G. Shin and N.D. McKay, *IEEE Transactions on Automatic Control*, Vol. AC-32, No. 12, pp. 1044-1054, December 1987.

10. "Coordination of Dual Robot Arms using Redundancy," I.H. Suh and K.G. Shin, *IEEE Journal of Robotics and Automation*, (in press).

11. "PAR: A CSG-based Unique Representation Scheme for Rotational Parts," Y.C. Lee, K.F. Jea, *IEEE Trans. on Systems, Man and Cybernetics*, Vol. SMC-17, No. 11, Nov/Dec 1987.

3.2 Conference Presentations and Papers

1. "Integrated Manufacturing Software in Ada," R.A. Volz, A.W. Naylor, *Conference on Computer Controlled Mechanics*, Stockholm, Sweden, May 26, 1988.

2. "Improved Decision Trees: A Generalized Version of ID3," Jie Cheng, Usama M. Fayyad, Keki B. Irani, Zhaogang Qian, *Proceedings of the Fifth International Conference on Machine Learning*, Ann Arbor, MI, June 1988.

3. "Learning Control Information in Rule-Based Systems: A Weak Method," Usama M. Fayyad, Kristina E. Van Voorhis, Mark D. Wiesmeyer, *Proceedings of the Fourth Conference on Artificial Intelligence*, San Diego, CA, March 14-18, 1988.

4. "A Qualitative Approach for Recovering Relative Depths in Dynamic Scenes," S.M. Haynes, Ramesh Jain, *Proceedings of Workshop on Computer Vision*, pp. 66-71, Miami Beach, FL, November 30 - December, 2, 1987

5. "Symbolic Surface Descriptors for 3-Dimensional Object Recognition," R. Jain, T. Sripradisvarakul, N. O'Brien, *Proceedings of SPIE*, vol. 754, pp. 82-91, January 1987.

6. "Implementation of Task Types in Distributed Ada," P. Krishnan, R.A. Volz and R.J. Theriault, *2nd International Workshop on Real Time Ada Issues*, Devon, England, 1-3, June 1988.

7. "Coordination of Dual Robot Arms Using Kinematic Redundancy", I.H. Suh and K.G. Shin, *IEEE Int'l Conf. on Robotics and Automation*, pp. 504-509, April, 1988.

8. "A Probabilistic Approach to Collision-Free Path Planning for Industrial Robots", S. Jun and K. Shin, *IEEE Int'l Conf. on Robotics and Automation*, pp. 220-225, April, 1988.

9. "Incremental Inference: Spatial Reasoning within a Blackboard Architecture," Terry E. Weymouth, *Workshop on Spatial Reasoning and Multi-Sensor Fusion*, October 1987

10. "Design and Motion Constraints of Part-Mating Planning in the Presence of Uncertainties," J. Xiao, R.A. Volz, *IEEE International Robotics and Automation Conference*, September 1987.

11. "Region Grouping from a Range Image," J.H. Han, R.A. Volz, *IEEE Computer Society Conference on Computer Vision & Pattern Recognition*, June 1988.

12. "Robot Path Planning with a Weighted Distance-Safety," S.-H. Suh and K.G. Shin, *Proc. Conf. on Decision and Control*, pp. 634-641, December 1987.

13. "Parallel Branch and Bound Algorithms on Hypercube Multiprocessors," T.S. Abdelrahman, T.N. Mudge, *Third Conference on Hypercube Computers and Applications*

14. "Subgoal Ordering and Goal Augmentation for Heuristic Problem Solving," Keki B. Irani, Jie Cheng, *IJCAI-87*, Milan, Italy.

15. "Motion Stereo and Ego-Motion Complex Logarithmic Mapping," S.L. Bartlett, R. Jain, *SPIE Technical Symposium Southeast on Optics, Electro-Optics, and Sensors*, April 4-8, 1988, Orlando, FL

16. "Wide Base-Line Dynamic Stereo: Approximation and Refinement," T.E. Weymouth, S. Moezzi, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), Ann Arbor, Michigan, June 5-9, 1988

17. "A New CSG Tree Reconstruction Algorithm for Feature Unification," Y.C. Lee, K.F. Jea, *Proc. 1988 ASME Computers in Engineering Conference*, July 31 - August 03, 1988, San Francisco, CA.

18. "Robot Trajectory Tracking with Self-Tuning Predicted Control," X. Cui, K.G. Shin *Proc. 1988 American Control Conference*, Vol. 1, June 15-17, 1988, Atlanta, Georgia.

19. "Effects of Computing Time Delay on Real-Time Control Systems," X. Cui, K.G. Shin *Proc. 1988 American Control Conference*, Vol. 1, June 15-17, 1988, Atlanta, Georgia.

3.3 Under Review

Journals

1. "Design and Motion Constraints of Part-Mating Planning in the Presence of Uncertainties," J. Xiao, R.A. Volz, *submitted IEEE Journal of Robotics and Automation*

Conferences

3.4 Papers in Progress

Journals

1. Paul G. Gottschalk and Trevor Mudge, "Efficient Encoding of Local Shape: Features for 3-d Object Recogntion", *SPIE Conference 1002: Intelligent Robots and Computer Vision.*

Conferences

4 Personnel

4.1 Faculty

There are many ways in which a faculty member or student may be supported by a contract of this type. The names listed on the form DD1473 include only those faculty who received direct salary support from the contract. However, in addition, some faculty received support indirectly through support of their graduate or post doctoral students. This section lists faculty who have either directly received support for a portion of their salaries or received support for one or more of their graduate students.

Professors

Keki B. Irani	Electrical Engineering and Computer Science
Ramesh Jain	Electrical Engineering and Computer Science
Kang G. Shin	Electrical Engineering and Computer Science
Richard A. Volz	Electrical Engineering and Computer Science

Associate Professors

Trevor N. Mudge	Electrical Engineering and Computer Science

Assistant Professors

Y.-C. Lee	Electrical Engineering and Computer Science
T.E. Weymouth	Electrical Engineering and Computer Science

4.2 Students

The number of students who have received support during the second year of this contact exceeds the number of "standard" student appointments indicated in the proposal. There are two primary reasons for this. First, and most importantly, we were able to leverage the student funding to increase the student involvement. In a number of cases we were able to obtain partial fellowship support for the student from internal funds, thus reducing the cost per student and increasing the number of students we could involve in the project. Similarly some students had partial outside fellowship support and only needed supplementary support to be able to participate in the project. The involvement of students with fellowship support in the project is deemed an acceptable activity since the work being performed is generally part of the student's dissertation work. Secondly, through natural attrition, graduation, and staff changeover, some replacement of the students supported occurred during the year. The students listed below, then, include all of the students who have received support during the contract year.

Sandra Bartlett	CSE[10]Ph.D. expected 1988-89
Fan Jiang	ECE[11]Ph.D. expected 1988-89
Sung Taeg Jun	EECS[12]Ph.D. expected 1988-89
Oyekunle Olukotun	EECS Ph.D. expected 1988-89
Thawach Sripradisvarakul	CICE[13]Ph.D. expected 1988-89
Jing Xiao	CICE Ph.D. expected 1989
Jie Cheng	CICE Ph.D. expected 1988-89
C.-H. Cho	CSE Ph.D. expected 1989-90
Usama Fayyad	CSE Ph.D. expected 1989-90
Paul Gottschalk	CICE Ph.D. expected 1988-89
Paddy Krishnan	CSE Ph.D. expected 1989

4.3 Degrees Awarded

Tarek Abdel-Rahman	Ph.D. in – CICE
Rajiv Desai	Ph.D. in – Mechanical Engr.
Jan Wolter	Ph.D. in – CICE

[10]Computer Science and Engineering
[11]Electrical and Computer Engineering
[12]Electrical Engineering and Computer Science
[13]Computer, Information, and Control Engineering

4.4 Graduate Student Placement

1982-1983

Gary Swanson, Ph.D.	Lincoln Labs
Henry Chu, M.S.	U of M Ph.D. Program
Paul Eichel, M.S.	U of M Ph.D. Program
Stacy Leon, M.S.	U of M Ph.D. Program
Mike Steigerwald, M.S.	U of M Ph.D. Program
Jan Wolter, M.S.	U of M Ph.D. Program
Brian Dent, M.S.	ADAPCO, Melville, New York
Myrung Chung, Ph.D.	Korean Institute of Technology
Dean Askounis, M.S.	FMC Robots, Troy, Michigan

1983-1984

Thomas Wielenga, Ph.D.	M.D.I., Ann Arbor, Michigan
Joseph Pincu, M.S.	U of M Ph.D. Program
Daniel Barash, M.S.	Harris Corporation, Washington D.C.
Paul Firehammer, M.S.	unknown
Nirwan Ansari, M.S.	U of M Ph.D. Program
David Gal, M.S.	R.O.L.M. in California
Mark Epstein, M.S.	IBM, Boca Raton, Florida
Neil Nathason, M.S.	IBM, Toledo, Ohio
Chi-Chan Tsang, M.S.	Computerized Office Systems, Ann Arbor
Glen Healy, B.S.	Stanford Graduate Program
Elesh Shah, M.S.	unknown
Jerry Turney, M.S.	U of M Ph.D. Program

1984-85

D. Beard, Ph.D.	University of North Carolina
B. Lee, Ph.D.	Purdue University
N. McKay, Ph.D.	General Motors Research Laboratory

J. Ha, Ph.D.	General Motors Research Laboratory
P. Eichel, Ph.D.	Sandia Laboratory
J. Lah, Ph.D.	Intel Corporation
S. Yoo, Ph.D.	Seoul National University
D. Shin, Ph.D.	University of Connecticut
P. Bixel, M.S.	General Electric
K. Lloyd, M.S.	Bell Laboratories
R. Rubinfeld, B.S.	University of California, Berkeley

1985-86

Paul Besl, Ph.D.	General Motors Tech Center
Nabil Chalhoub, Ph.D.	University of Reno
Il Hyun Choi, Ph.D.	IBM - Virginia
Kukjin Chun, Ph.D.	University of Washington
H.P. Huang, Ph.D.	National Taiwan University
Brian Schipper, M.S.E.	unknown
Jerry Turney, Ph.D.	KMS Fusion Inc.

1986-87

Dan Johnson, Ph.D.	Martin Marietta, Baltimore
Murtaza Doctor, M.S.	MIT (Ph.D. Program)

1987-88

Tarek Abdel-Rahman, Ph.D.	University of Iowa, Computer Science Dept.
Rajiv Desai, Ph.D.	JPL, Pasadena, California
Nancy O'Brien Byrd, M.S.	General Dynamics
Jan Wolter, Ph.D.	Texas A&M University

4.5 Permanent Staff

Research Engineer	**Systems Programmers**	**Technician**
Al Dobryden	Brent Harper	Rob Giles
	Ron Theriault	

Administrative Assistant	**Secretaries**
Janice Ransom	Dolores Bolsenga
	Marianne Moylan
	Elizabeth Olsen

5 Coupling Activities

An important aspect of our program is the development of strong coupling activities both with and without the program. Four categories of interaction are being pursued:

- interaction among people participating in the project
- interactions with other university scientific groups
- interactions with industry
- interactions with the Air Force

The following sections describe the coupling activities which have taken place during the past year.

5.1 Intra Project Interactions

There are regular technical discussion meetings among members of several of the research groups occurring on a weekly basis. Membership in more than one group is common, particularly among the faculty concerned. Also, most of the faculty are members of several doctoral committees on topics outside of their own immediate research activities. This has proven to be a useful form of information diffusion.

5.2 University Interactions

Interactions with people from other universities have taken place primarily through seminars, both those given at the University of Michigan, and those given by Michigan people at other universities.

Outside Seminar Speakers

- Dr. J. Fainman, University of California - San Diego
  "*Optical Signal Processing,*" *January* 1988.
- Professor D. Mote, University of California - Berkeley
  "Vibration and Stability of Saw Blades," March 1988.
- Professor S. Malkin, University of Massachusetts - Amherst
  "Grinding Theory Optimization," March 1988.

- Dr. M. Shpitalni, University of California - Santa Barbara
  "Solid Modeling for Assembly and Machining," April 1988.

Seminars at other Universities given by Michigan faculty

- Professor R.A. Volz - Texas A&M University
  "Research in Integrated Manufacturing"
- Professor R.A. Volz - Royal Technical Institute in Stockholm Sweden
  "Integrated Manufacturing Software in Ada"
- Professor R. Jain - University of California - Santa Barbara,
  "CAD Based Object Recognition"
- Professor R. Jain - University of Central Florida - Orlando
  "CAD Based Object Recognition"
- Professor K.G. Shin - Pennsylvania State University
  "Integrated Multiple Robot Systems," *Computer Networking for Manufacturing Systems*, November 1987.

5.3 Interactions with Industry

Seminars and Briefings for Industry

- Professor K.B. Irani - General Motors Research Labs.
- Professor R. Jain - General Dynamics, GM, Advanced Engineering Staff, ERIM
- Professor R.A. Volz - General Motors, Geospectra, Verdix
- Professor K.G. Shin - General Motors and Ford, NASA
- Professor T.E. Weymouth - General Dynamics

Seminars at Michigan by Industrial People

- Dr. Henry W. Stoll, Industrial Technology Institute
  "New Approaches to Product/Process Design," September 1987.

- Mr. Edward Miller, National Center for Manufacturing Sciences
  "The Future of Manufacturing Sciences," December 1987.

- Dr. M. Wozni, DMCE Division Director, National Science Foundation
  "Computer Graphics and CAD/CAM," February 1988.

Private technical discussions with industry

- Professor R. Jain has had regular discussions with ERIM, Ford, Hitachi, Dana

- Professor T.N. Mudge - Vicom, KMS Fusion, Astronautics of America, and Applied Dynamics International

- Professor R.A. Volz - General motors, Ford Aerospace, Grumman Aerospace, Geospectra, ASEA, KMS Fusion, Deneb Robotics, Verdix, DDCI, Industrial Technology Institute

- Professor Y.C. Lee - General Motors, Harris

- Professor T.E. Weymouth - General Dynamics

- Professor K.G. Shin - GM, Ford, NASA, Ford Aerospace

Faculty and Students Giving Seminars within the University

- "Tele-Autonomous System Update"
  Professor R.A. Volz, October 1987.

- "Feedback Regulation of the Position and Contact Force of a Two-Dimensional Mechanism on a Given Planar Contour"
  Professor N.H. McClamroch, October 1987.

- "3-D Sensing Using Fiber Optics"
  Professor L. Damm, October 1987.

- "Mobile Robot Navigation"
  J. Borenstein, October 1987.

- "Knowledge-Based Systems in a General Cognitive Architecture"
  Professor J. Laird, October 1987.

- "Dynamics of a Radially Rotating Beam with Impact"
  Ahmet Yigit and Professor A. Ulsoy, October 1987.

- "Effects of Computing Time Delay on Real-Time Control Systems"
  Xianghong Cui, June 1988.
- "An Efficient Algorithm for the Adaptive Control of a Manipulator"
  Professor M.W. Walker, October 1987.
- "Experimental Results in Robot Control"
  Professor M.W. Walker, May 1988.
- "Automating Knowledge Acquisition: An Algorithm for Generating Improved Decision Trees in Multiple Concept Learning"
  Professor K.B. Irani and U. Fayyad, May 1988.
- "Using Soar as the Basis for an Integrated Autonomous Agent."
  Professor J. Laird, May 1988.
- "Simulation of Robots"
  Professor R.M. Howe, May 1988.
- "Automated Detection of Clearance in Mechanical Systems"
  Professor J.L. Stein, May 1988.
- "Real-Time Obstacle Avoidance for Fast Mobile Robots"
  J. Borenstein and Professor Y. Koren, May 1988.
- "Adaptive Control of a Production Queueing System"
  Professor N.H. McClamroch and C.-P. Lee, May 1988.
- "Designing for Manufacture"
  Professor S.W. Thomas, May 1988.
- "An Intelligent CAD Modeler for the Conceptual Design Phase"
  Mark Jakiela, May 1988.
- "Experimental Validation of Dynamic Models for Flexible Robot Arms"
  Professor A.G. Ulsoy, May 1988.
- "Object Recognition"
  Paul Gottschalk, October 1987.
- "A Review of Research in Knowledge-Based Computer Vision"
  Professor T.E. Weymouth, May 1988.
- "A Visual Language for Interactions with a Vision System"
  Professor R. Jain and R. Rao, October 1987.

- "Minimal Motion Stereo"
  Kurt Skifstad and Professor R. Jain, May 1988.
- "Design and Motion Constraints of Robot Part-Mating Planning in the Presence of Uncertainties"
  Jing Xiao, May 1988.
- "A Probabilistic Approach to Collision-Free Path Planning for Industrial Robot"
  Sungtaeg Jun and Professor K.G. Shin, May 1988.
- "Cyclops: A 3-D Object Recognizer"
  Paul G. Gottschalk and Professor T.N. Mudge, May 1988.
- "ESQL: An Enhanced but Simple Query Language for a Semantic Data Model"
  Professor Y.-C. Lee, October 1987.

5.4 Government Interaction

- Professor Y.C Lee - UCLA/NSF Workshop on Features in Design and Manufacturing, February 1988
- Professor K.G. Shin - NSF, "Systems Integration to NSF"
- Professor K.G. Shin - NASA, "Real-Time Networking"
- Professor R.A. Volz - NASA/Ames, "An Introduction to Tele-autonomous Systems," November 1987
- Professor R.A. Volz - NASA/USRA, "Remote Coaching, a New Application of Tele-Autonomous Systems," March 1988
- Professor R.A. Volz - Institute for Defense Analysis, "Real-Time Distributed Ada," June 1988
- Professor R.A. Volz - Service on the following advisory boards:
  - Aerospace Safety Advisory Panel – Congressional oversight committee for NASA on Safety
  - The Ada Board – Advisory board to the Ada Joint Program Office
  - Ad hoc advisory panels for NASA on Tele-robotics and the Pathfinder program
  - IDA advisory workshop on new issues in Ada
  - Ad hoc DARPA planning committee on new initiatives in manufacturing

A Distribution List of The Report

University and Research Institutes

Professor Thomas Binford
Computer Science Dept.
Stanford University
Stanford, CA 94305

Professor Robert McGhee
Dept. of Electrical Engineering
Ohio State University
2015 Neil Avenue
Columbus, OH 43210

Professor Raj Reddy
Computer Science Dept. & Robotics
Carnegie Mellon University
Schenley Park
Pittsburgh, PA 15213

Professor John Hollerbach
MIT, AI Lab
545 Technology Square
RM NE 43-771
Boston, MA 02139

Professor Richard Paul
Dept. of Computer Information
and Sciences
University of Pennsylvania
N60 Town Bldg.
Philadelphia, PA 19104

Professor Roger Nagel
Lehigh University
Bethlehem, PA 18015

Professor Georges Saridis
ECSE
Rensselaer Polytechnical Institute
Troy, NY 12181

Professor Delbert Tesar
Dept. of Mechanical Engineering
University of Texas
Austin, Texas

Dr. David Whitney
Draper Laboratories
555 Technology Square
Massachusetts Institute of Technology
Cambridge, MA 02139

Dr. David Nitzen
Director of Robotics Research
Stanford Research Institute
333 Ravenswood Avenue
Menlo Park, CA 94025

Dr. Robert Bolles
Stanford Research Institute
333 Ravenswood Avenue
Menlo Park, CA 94025

Dr. William Brown
ERIM
P.O. Box 8618
Ann Arbor, MI 48107

Dr. Azriel Rosenfeld
Research - Room 4115C
Computer & Space Sciences Bldg.
University of Maryland
College Park, MD 20742

Dr. Geoffrey Boothroyd

Mechanical Engineering
University of Massachusetts
Amhurst, MA 01003

Dr. Don Falkenburg
Industrial Technology Institute
Beal Avenue
Ann Arbor, MI 48109

Government

Dr. James Albus
Room A127 Bldg 220
National Bureau of Standards
Washington D.C. 20234

Dr. Turkovich
Program Director
Automation and System Integration
Division of Design, Manufacturing
and Computer Engineering
National Science Foundation
1800 G. Street N.W.
Washington D.C. 20550

Mr. Bernie Chern
Division of Electrical, Computer &
System Engineering
Room 1101
National Science Foundation
Washington D.C. 20550

Captain Robert Delyser
DFEE
U.S. Air Force Academy
Colorado 80840

Dr. Charles Holland
Head
Dept. of Navy
Information Sciences Division
Office of Naval Research
800 N. Quincy
Arlington, VA 22217

Dr. Vincent Russo
Wright Patterson Air Force Base
AFWAL/MLTC
Wright Patterson, OH 45433

Mr. Thomas Lagnese
Project Director
Computer Integrated
Manufacturing Branch
Department of Air Force
Air Force Wright Aeronautical Lab
Wright Patterson Air Force Base,
Ohio 45433

Mr. Michael Hitchcock
Computer Integrated
Manufacturing Branch
Department of Air Force
Air Force Wright Aeronautical Lab
Wright Patterson Air Force Base,
Ohio 45433

Mr. William M. Spurgeon
University of Michigan at Dearborn
Dearborn, Michigan

Mr. G. Ronald Green
AIRMICS
115 O'Keefe Building
Georgia Institute of Technology

Atlanta, GA 30332-0800

Mr. Ted J. Brandewie
Computer Integrated
Manufacturing Branch
Department of Air Force
Air Force Wright Aeronautical Lab
Wright Patterson Air Force Base,
Ohio 45433

Dr. Lee Holcomb
NASA Headquarters
Washington DC, 20546

Major James Crowley
Directorate of Mathematics
and Information Sciences
AFOSR
Bolling Air Force Base
Washington, D.C. 20332-6448

Industry

Mr. Mark Hornick
ASEA
Industrial Robot Division
16250 W. Glendale
New Berlin, WI 53151

Dr. Gordon English
General Dynamics
Land Systems Division
P.O. Box 2074
Warren, MI 48090

Mr. Don Dresselhouse
General Dynamics
Land Systems Division
P.O. Box 1902
Warren, MI 48092

Dr. Steve Holland
Mr. Frank DiPietro
Mr. Gerald Elson
Mr. Gabe Tiberio
Mr. Richard Beecher
General Motors
General Motors Research Laboratories
Warren, MI 48090

Mr. Pete Matheson
Intel Corporation
5200 N.E. Elam Young Park Way
Hillsboro, OR 97123

Mr. J.L. Escover
Mr. Steve DeBrock
Lockheed
Missiles and Space Company, Inc.
P.O. Box 1700
Austin, TX 78760

Mr. Dwight Carlson
Perceptron
23920 Freeway Park Drive
Farmington Hills, MI 48024

Mr. Clifford T. Anglewicz
Volvo of America Corporation
Volvo Automated Systems Division
40712 Brentwood Drive
Sterling Heights, MI 48078

Dr. Gale Cutler

Whirlpool
Monte Road
Benton Harbor, MI 49022

Dr. John Klein
IBM
1798 NW 40th Street
Boca Raton, FL 33432

Mr. Bruce Haupt
IBM
951 N.W. 51st.
Boca Raton, FL 33432

Dr. Michael Wesley
IBM
T.J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

Mr. William Hollenback
IBM
Neighborhood Rd.
Kingston, NY 12401

Mr. George Leeman
IBM
Thomas J. Watson Research Ctr.
P.O. Box 704
Yorktown Hts, NY 10598

Mr. James Lowrie
Advanced Automation
Technology Section
Martin Marietta Aerospace
P.O. Box 179
Denver, CO 80209

Mr. Donald E. Waters
Program Manager
Technical Strategy Center
Honeywell, Inc.
1700 W. Highway 36
St. Paul, MN 55113

Interim Dean D.E. Atkins (3)
University of Michigan
2401 EECS Bldg.
1301 Beal Ave.
Ann Arbor, MI 48109-2116

Professor R. Carignan (2)
University of Michigan
2306 EECS Bldg.
1301 Beal Ave.
Ann Arbor, MI 48109-2116

Professor Lynn Conway (1)
College of Engineering
University of Michigan
2307 EECS Bldg.
Ann Arbor, MI 48109-2116

Professor Elmer G. Gilbert (1)
Dept. of Aerospace Engineering
The University of Michigan
Ann Arbor, MI 48109-2140

Professor Keki B. Irani (1)
Dept. of Electrical and Computer Science
University of Michigan
1301 Beal Ave.
Ann Arbor, MI 48109-2122

Professor Walton M. Hancock (1)

Associate Dean for CRIM
College of Engineering
The University of Michigan
1101 Beal Ave., Rm 170 ATL Bldg.
Ann Arbor, MI 48109-2110

Professor Ramesh Jaini (1)
University of Michigan
AI Lab., 120 ATL Bldg.
1101 Beal Ave.
Ann Arbor, MI 48109-2110

Professor Yoram Koren (3)
University of Michigan
1101 Beal Ave., Rm 110 ATL Bldg.
Ann Arbor, MI 48109-2110

Professor C.S.G. Lee (2)
School of Electrical Engineering
Purdue University
West Lafayette, IN 47907-0501

Professor Y.C. Lee (1)
University of Michigan
1101 Beal Ave., 128 ATL Bldg.
Ann Arbor, MI 48109-2110

Professor N.H. McClamroch (1)
Dept. of Aerospace Engineering
University of Michigan
Rm. 311 AEB
Ann Arbor, MI 48109-2140

Professor T.N. Mudge (1)
University of Michigan
1101 Beal Ave., Rm. 134 ATL Bldg.
Ann Arbor, MI 48109-2110

Professor Jeff Stein (1)
Dept. of MEAM
University of Michigan
2250 GG Brown
Ann Arbor, MI 48109-2125

Professor A. Galip Ulsoy (1)
Dept. of Mechanical Engineering
University of Michigan
2250 GG Brown
Ann Arbor, MI 48109-2125

Professor Michael W. Walker (1)
University of Michigan
1101 Beal, RM 126 ATL Bldg.
Ann Arbor, MI 48109-2110

Professor Richard A. Volz (3)
238 Zachry Engineering
Computer Science Department
Texas A&M University
College Station, Texas 77843-3112

Professor Terry E. Weymouth (1)
University of Michigan
1101 Beal Ave., 164 ATL Bldg.
Ann Arbor, MI 48109-2110

Professor Ken D. Wise (1)
1246 EECS Bldg.
1301 Beal Ave.
University of Michigan
Ann Arbor, MI 48109-2122

B Robotics Industrial Affiliates Members List

Deneb Robotics

Mr. R. Mahajan
Director Sales and Marketing
1120 East Long Lake Road, Suite 200
Troy, MI 48098
313 689-7763

Digital Equipment Corp.

Thomas Zavorski
4 Omni Drive
CTS,C6
Chelmsford, MA 01824
617-250-3525

Wm. Mulcahy
20 Alpha Road, ICO/E25
Chelmsford, MA 01824
617 250-2429

General Motors

Mr. Steven W. Holland
Assistant Head
Computer Science Department
General Motors Res. Lab
Warren, MI 48090-9057
313-986-1510

Ms. Kelly Talaki
General Motors Technical Center
Environment Activities Bldg.
30400 Mound Road
Mail Stop/ First Floor West
Warren, MI 48090-9015
313 947-1209

Grumman Aerospace Corporation

Mr. Gil Carpenter
Aircraft System Division
Mail Station A4/35
Bethpage, NY 11714
516-575-2203

Mr. Warren Marx
Mail Station A4/12
516-575-6397

Mr. Eric Byler
Mail Station B18-002
516-575-7979

Mr. Ronald Spencer
Mail Station A4/35
Bethpage, NY 11714-3580
516-575-2579

Lockheed

Mr. Richard C. Movich
Robotics/CIM Program Manager
Dept. 47-11, Bldg. 152, Plant B-1
Lookheed-California Co.
Burbank, CA 91520-4711
213-847-8524

Mr. Walt Plumley
Lockheed Missiles and Space Co., Inc.
Department 41-11
Building 1, Zone 102
86 South Cobb Drive
Marietta, GA 30063
404-424-3796

Mr. S.C. DeBrock
Lockheed Missiles and Space Co., Inc.
Department 53-40
Building 580

1111 Lockheed Way
Sunnyvale, CA 94086
408-743-1571

Mr. J.L. Escover
Lockheed Missiles and Space Co., Inc.
Dept. T3-35, Bldg. 30-D
Missiles and Space Co., Inc.
2124 East St. Elmo Street
P.O. Box 1700
Austin, Texas 78760
512-362-7303

Westinghouse Corporation

Edward P. Liscio
Artificial Intelligence
335 E. Expo-Mart
105 Mall Blvd.
Monroeville, PA 15146
412 374-7529

Don Thomas
Program Development Manager
P.O. Box 355
Pittsburgh, PA 15230-355
412 722-5439

Dennis Richardson
Expo-Mart 339E
105 Mall Blvd.
Monroeville, PA 15147
412 374-7190

*U.S.Government Printing Office: 1989 — 748-056/24094